

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号

特許第6567100号  
(P6567100)

(45) 発行日 令和1年8月28日(2019.8.28)

(24) 登録日 令和1年8月9日(2019.8.9)

(51) Int. Cl.	F 1
HO 4 N 19/122 (2014.01)	HO 4 N 19/122
HO 4 N 19/70 (2014.01)	HO 4 N 19/70
HO 4 N 19/176 (2014.01)	HO 4 N 19/176
HO 4 N 19/136 (2014.01)	HO 4 N 19/136
HO 4 N 19/186 (2014.01)	HO 4 N 19/186

請求項の数 12 (全 37 頁)

(21) 出願番号	特願2018-9529 (P2018-9529)	(73) 特許権者	000001007 キヤノン株式会社 東京都大田区下丸子3丁目30番2号
(22) 出願日	平成30年1月24日(2018.1.24)	(74) 代理人	100126240 弁理士 阿部 琢磨
(62) 分割の表示	特願2015-533379 (P2015-533379) の分割	(74) 代理人	100124442 弁理士 黒岩 創吾
原出願日	平成25年9月27日(2013.9.27)	(72) 発明者	クリストファー ジェームズ ロゼワーン オーストラリア国 2113 ニューサウス ウェールズ州 ノースライド、トーマス ホルト ドライブ 1 キヤノン イン フォメーション システムズ リサーチ オーストラリア プロプライエタリー リ ミテッド内
(65) 公開番号	特開2018-101984 (P2018-101984A)		
(43) 公開日	平成30年6月28日(2018.6.28)		
審査請求日	平成30年1月24日(2018.1.24)		
(31) 優先権主張番号	2012232992		
(32) 優先日	平成24年9月28日(2012.9.28)		
(33) 優先権主張国・地域又は機関	オーストラリア(AU)		

最終頁に続く

(54) 【発明の名称】 符号化単位の変換単位を符号化および復号する方法、装置、およびシステム

(57) 【特許請求の範囲】

【請求項1】

ビデオビットストリームから、4:2:2フォーマットの符号化単位内の変換単位に含まれる、2つの色差チャンネルにそれぞれ関連付けられた2つの色差残差係数配列を復号し、前記復号された色差残差係数配列を周波数領域から空間領域の色差残差サンプルへ変換する方法であって、

符号化単位のサイズおよび前記符号化単位内の変換単位の階層レベルに従って、正方形ブロックで表される前記変換単位の変換サイズを決定するステップと、

各色差チャンネルにおいて、

前記ビデオビットストリームから、前記変換単位が有する2つの色差残差係数配列のそれぞれに対応する2つの符号化ブロックフラグ値を復号するステップと、

前記2つの符号化ブロックフラグ値のうちの対応する符号化ブロックフラグ値に従って、前記2つの色差残差係数配列の各々を前記ビデオビットストリームから復号するステップと、

前記決定された変換サイズに対応する正方形ブロックのための変換を、前記復号された色差残差係数配列の各々に対して適用することにより、色差残差サンプルを生成するステップと、

を備え、

前記2つの色差残差係数配列の各々を前記ビデオビットストリームから復号するステップにおいて、前記2つの色差残差係数配列の各々に対して同じスキャンパターンを適用す

ることにより、正方形ブロックで表される前記決定された変換サイズの変換単位を生成することを特徴とする方法。

【請求項 2】

空間領域の色差残差サンプルを周波数領域への変換を行うことにより、2つの色差チャンネルのそれぞれにおいて、4:2:2フォーマットの符号化単位内の変換単位に含まれる2つの色差残差係数配列としてビットストリームに符号化する方法であって、

各色差チャンネルにおいて、

前記符号化単位のサイズおよび前記符号化単位内の前記変換単位の階層レベルに従って決定されるサイズの正方形ブロックのための変換を、前記変換単位の2セットの色差残差サンプルの少なくとも1つに対して適用することにより、前記2つの周波数領域の色差残差係数配列を生成するステップと、

前記変換単位の2つの色差残差係数配列にそれぞれ対応する2つの符号化ブロックフラグ値および前記2つの色差残差係数配列を前記ビデオビットストリームに符号化するステップと、を備え、

前記2つの色差残差係数配列を生成するステップにおいて、前記変換によって生成される2セットの残差係数に対して同じスキャンパターンを適用することにより、前記2つの色差残差係数配列を生成することを特徴とする方法。

【請求項 3】

請求項 1 において、前記符号化ブロックフラグ値は、対応する色差残差係数配列にゼロではない残差係数が含まれるか否かを示すことを特徴とする方法。

【請求項 4】

請求項 2 において、前記符号化ブロックフラグ値は、対応する色差残差係数配列にゼロではない係数が含まれるか否かを示すことを特徴とする方法。

【請求項 5】

ビデオビットストリームから、4:2:2フォーマットの符号化単位内の変換単位に含まれる2つの色差チャンネルにそれぞれ関連付けられた2つの色差残差係数配列を復号し、復号された複数の色差残差係数配列を周波数領域から空間領域の色差残差サンプルへ変換する復号装置であって、

符号化単位のサイズおよび前記符号化単位内の変換単位の階層レベルに従って、正方形ブロックで表される前記変換単位の変換サイズを決定するステップと、

各色差チャンネルにおいて、

前記ビデオビットストリームから、前記変換単位が有する2つの色差残差係数配列のそれぞれに対応する2つの符号化ブロックフラグ値を復号する手段と、

前記2つの符号化ブロックフラグ値のうちの対応する符号化ブロックフラグ値に従って、前記2つの色差残差係数配列の各々を前記ビデオビットストリームから復号する手段と、

前記決定された変換サイズに対応する正方形ブロックのための変換を、前記復号された色差残差係数配列の各々に対して適用することにより、色差残差サンプルを生成する手段と、

を備え、

前記2つの色差残差係数配列の各々を前記ビデオビットストリームから復号する手段において、前記2つの色差残差係数配列の各々に対して同じスキャンパターンを適用することにより、正方形ブロックで表される前記決定された変換サイズの変換単位を生成することを特徴とする復号装置。

【請求項 6】

請求項 5 において、前記符号化ブロックフラグ値は、対応する色差残差係数配列にゼロではない残差係数が含まれるか否かを示すことを特徴とする復号装置。

【請求項 7】

請求項 5 または請求項 6 の復号装置の処理を実行させるためのプログラム。

【請求項 8】

10

20

30

40

50

請求項7のプログラムを記憶したコンピュータ可読な記憶媒体。

【請求項9】

空間領域の色差残差サンプルを周波数領域への変換を行うことにより、2つの色差チャンネルのそれぞれにおいて、4:2:2フォーマットの符号化単位内の変換単位に含まれる2つの色差残差係数配列としてビットストリームに符号化する符号化装置であって、

各色差チャンネルにおいて、

前記符号化単位のサイズおよび前記符号化単位内の前記変換単位の階層レベルに従って決定されるサイズの正方形ブロックのための変換を、前記変換単位の2セットの色差残差サンプルの少なくとも1つに対して適用することにより、前記2つの周波数領域の色差残差係数配列を生成する手段と、

前記変換単位の2つの色差残差係数配列にそれぞれ対応する2つの符号化ブロックフラグ値および前記2つの色差残差係数配列を前記ビデオビットストリームに符号化する手段と、を備え、前記2つの色差残差係数配列を生成する手段において、前記変換によって生成される2セットの残差係数に対して同じスキャンパターンを適用することにより、前記2つの色差残差係数配列を生成することを特徴とする符号化装置。

10

【請求項10】

請求項9において、前記符号化ブロックフラグ値は、対応する色差残差係数配列にゼロではない残差係数が含まれるか否かを示すことを特徴とする符号化装置。

【請求項11】

請求項9または請求項10の符号化装置の処理を実行させるためのプログラム。

20

【請求項12】

請求項11のプログラムを記憶したコンピュータ可読な記憶媒体。

【発明の詳細な説明】

【関連出願の相互参照】

【0001】

本出願は、2012年9月28日に提出されたオーストラリア国特許出願第2012232992号の出願日の権利を米国特許法第119条の下で主張し、本明細書において十分に示されるようにその全体を参考文献として援用する。

【技術分野】

【0002】

本発明は、一般的に、デジタルビデオ信号処理に関し、特に、変換単位(TU)の残差係数を符号化し復号化するための方法、装置およびシステムに関する。ここで、変換単位(TU)は、1つ以上の変換単位(TU)を含み、4:2:2色差フォーマットを含む複数の色差フォーマット用に構成されてもよい。

30

【背景技術】

【0003】

現在、ビデオデータの伝送および記憶のためのアプリケーションを含む、ビデオ符号化のための多くのアプリケーションが存在する。多くのビデオ符号化規格も開発されていて、他の規格も現在開発中である。ビデオ符号化の規格化における最近の展開は、「映像符号化共同研究部会」(JCT-VC)と呼ばれるグループの形成に結びついた。映像符号化共同研究部会(JCT-VC)は、映像符号化専門家グループ(VCEG)として知られている国際電気通信連合(ITU)の電気通信標準化部門(ITU-T)の第16研究グループQuestion6(SG16/Q6)のメンバーと、動画専門家グループ(MPEG)として知られている国際標準化機構/国際電気標準会議第一合同技術委員会/小委員会29/作業部会11(ISO/IEC JTC1/SC29/WG11)のメンバーとを含む。

40

【0004】

映像符号化共同研究部会(JCT-VC)は、「H.264/MPEG-4 AVC」として知られている現存するビデオ符号化規格よりも著しく性能が優れている新規のビデオ符号化規格を生み出すことを目標とする。H.264/MPEG-4 AVC規格は、

50

それ自体、従来のビデオ符号化規格（MPEG-4およびITU-T H.263）に対する大規模な改良である。開発中の新規のビデオ符号化規格は「高効率動画像符号化方式（HEVC）」と命名された。映像符号化共同研究部会JCT-VCは、また、リアルタイムまたは高フレームレートにおいて高分解能で動作する規格の実施態様をスケーリングする際に困難が生じる高効率動画像符号化方式（HEVC）のために提案された技術から生じる実施課題を検討している。1つの実施課題は、周波数領域と空間領域との間でビデオデータを変換するための複数「変換」サイズサポートするために用いられるロジックの複雑性およびサイズである。

【発明の概要】

【0005】

本発明は、既存の仕組みの1つ以上の短所を、実質的に克服するか、または少なくとも改善することを目的とする。

本開示の1つの態様によれば、ビデオビットストリームから、4:2:2フォーマットの符号化単位内の変換単位に含まれる、2つの色差チャンネルにそれぞれ関連付けられた2つの色差残差係数配列を復号し、前記復号された色差残差係数配列を周波数領域から空間領域の色差残差サンプルへ変換する方法であって、

符号化単位のサイズおよび前記符号化単位内の変換単位の階層レベルに従って、正方形ブロックで表される前記変換単位の変換サイズを決定するステップと、

各色差チャンネルにおいて、

前記ビデオビットストリームから、前記変換単位が有する2つの色差残差係数配列のそれぞれに対応する2つの符号化ブロックフラグ値を復号するステップと、

前記2つの符号化ブロックフラグ値のうちの対応する符号化ブロックフラグ値に従って、前記2つの色差残差係数配列の各々を前記ビデオビットストリームから復号するステップと、

前記決定された変換サイズに対応する正方形ブロックのための変換を、前記復号された色差残差係数配列の各々に対して適用することにより、色差残差サンプルを生成するステップと、

を備え、

前記2つの色差残差係数配列の各々を前記ビデオビットストリームから復号するステップにおいて、前記2つの色差残差係数配列の各々に対して同じスキャンパターンを適用することにより、正方形ブロックで表される前記決定された変換サイズの変換単位を生成することを特徴とする方法が提供される。

他の態様も開示される。

【図面の簡単な説明】

【0006】

ここで、本発明の少なくとも1つの実施形態が、以下の図面を参照しながら記載されることになる。

【図1】ビデオ符号化/復号化システムを示す概略ブロック図である。

【図2A】図1のビデオ符号化/復号化システム的一方または双方が実施されてもよい汎用コンピュータシステムの概略ブロック図を形成する。

【図2B】図1のビデオ符号化/復号化システム的一方または双方が実施されてもよい汎用コンピュータシステムの概略ブロック図を形成する。

【図3】ビデオエンコーダの機能モジュールを示す概略ブロック図である。

【図4】ビデオデコーダの機能モジュールを示す概略ブロック図である。

【図5A】フレームデータを表現するための色差フォーマットを概略的に図示する。

【0007】

【図5B】フレームデータを表現するための色差フォーマットを概略的に図示する。

【図6A】符号化単位の例示的な変換ツリーの概略図である。

【図6B】輝度サンプルグリッド上に配置された例示的な変換ツリーの概略図である。

10

20

30

40

50

【図 6 C】色差サンプルグリッド上に配置された例示的な変換ツリーの概略図である。

【図 7】例示的な変換ツリーの輝度チャンネルを表現するデータ構造の略図である。

【図 8】例示的な変換ツリーの色差チャンネルを表現するデータ構造を図示する。

【図 9 A】例示的な変換ツリーを符号化するビットストリーム構造を概略的に示す。

【図 9 B】例示的な変換ツリーを符号化するビットストリーム構造を概略的に示す。

【図 9 C】例示的な変換ツリーを符号化する代替的なビットストリーム構造を概略的に示す。

【図 9 D】例示的な変換ツリーを符号化する代替的なビットストリーム構造を概略的に示す。

【図 10】例示的な変換ツリーを符号化する方法を示す概略フローシートである。

10

【図 11】例示的な変換ツリーを復号化する方法を示す概略フローシートである。

【図 12 A】4 × 8 変換単位の残差スキャンパターンを概略的に図示する。

【図 12 B】4 × 8 変換単位の残差スキャンパターンを概略的に図示する。

【図 12 C】4 × 8 変換単位の残差スキャンパターンを概略的に図示する。

【発明を実施するための形態】

【0008】

添付の図面のいずれか 1 つ以上において、同一の参照符号を有するステップおよび / または特徴を参照する場合、それらのステップおよび / または特徴は、別段の意図が示されている場合を除いて、この記述にとって同一の機能または動作を有する。

【0009】

20

図 1 は、色差チャンネルのために変換単位を複数の変換に暗黙的に細分割 ( *inferred subdivision* ) することを表現するコード化構文要素のための技術を利用できるビデオ符号化 / 復号化システム 100 の関数モジュールを示す概略ブロック図である。システム 100 は、ソース装置 110 と宛先装置 130 とを含む。通信チャンネル 120 は、ソース装置 110 から宛先装置 130 に符号化映像情報を通信するために用いられる。いくつかのケースにおいて、ソース装置 110 および宛先装置 130 は、それぞれの携帯電話ハンドセットを備えてもよく、その場合、通信チャンネル 120 は、無線チャンネルである。他のケースにおいて、ソース装置 110 および宛先装置 130 は、ビデオ会議装置を備えてもよく、その場合、通信チャンネル 120 は、通常はインターネット接続などの有線チャンネルである。さらに、ソース装置 110 および宛先装置 130 は、テレビ放送、ケーブルテレビアプリケーション、インターネットビデオアプリケーションをサポートする装置を含み、符号化ビデオがいくつかの記憶媒体またはファイルサーバ上で取り込まれるアプリケーションを含む、任意の広範囲の装置を備えてもよい。

30

【0010】

図示するように、ソース装置 110 は、ビデオソース 112 と、ビデオエンコーダ 114 と、送信器 116 とを含む。ビデオソース 112 は、通常は、撮像センサ、非一時的な記録媒体に記憶された予め取り込まれたビデオシーケンス、またはリモート撮像センサから供給されたビデオなどの、取り込まれたビデオフレームデータのソースを備える。ビデオソース 112 のような撮像センサを含んでもよいソース装置 110 の例は、スマートフォン、ビデオカムコーダ、およびネットワークビデオカメラを含む。ビデオエンコーダ 114 は、ビデオソース 112 から取り込まれたフレームデータを符号化ビデオデータに変換し、図 3 を参照しながらさらに記述されることになる。符号化ビデオデータは、通常は、符号化ビデオ情報として通信チャンネル 120 を通じて送信器 116 によって送信される。また、符号化ビデオデータは、後で通信チャンネル 120 を通じて送信されるまでは、「フラッシュ」メモリまたはハードディスクドライブなどの何らかの記憶装置に記憶されることも可能である。

40

【0011】

宛先装置 130 は、受信器 132 と、ビデオデコーダ 134 と、ディスプレイ装置 136 とを含む。受信器 132 は、通信チャンネル 120 から符号化ビデオ情報を受信し、受信されたビデオデータをビデオデコーダ 134 に渡す。その後、ビデオデコーダ 134 は、

50

ディスプレイ装置 136 に対して復号化フレームデータを出力する。ディスプレイ装置 136 の例は、スマートフォン、タブレット型コンピュータ、コンピュータ用モニタ内の、またはスタンドアロン型テレビジョンセット内の、ブラウン管、液晶ディスプレイを含む。また、ソース装置 110 および宛先装置 130 の各々の機能性にとって、単一の装置で具体化されることも可能である。

#### 【0012】

上述の例示的な装置にもかかわらず、ソース装置 110 および宛先装置 130 の各々は、汎用コンピューティングシステム内で、通常は、ハードウェアおよびソフトウェアコンポーネントの組み合わせを通じて構成されてもよい。図 2A は、このようなコンピュータシステム 200 を図示しており、コンピュータシステム 200 は、コンピュータモジュール 201 と、キーボード 202、マウスポインタ装置 203、スキャナ 226、ビデオソース 112 として構成されてもよいカメラ 227、およびマイクロホン 280 などの入力装置と、およびプリンタ 215、ディスプレイ装置 136 として構成されてもよいディスプレイ装置 214、およびスピーカ 217 を含む出力装置とを含む。外部変調器復調器 (モデム) トランシーバ装置 216 は、接続 221 を介して通信ネットワーク 220 間で通信するためにコンピュータモジュール 201 によって用いられてもよい。通信チャンネル 120 に相当してもよい通信ネットワーク 220 は、インターネット、セルラ電気通信ネットワーク、またはプライベート WAN などのワイドエリアネットワーク (WAN) であってもよい。接続 221 が電話線である場合、モデム 216 は、従来の「ダイヤルアップ」モデムであってもよい。または、接続 221 が大容量 (例えばケーブル) 接続である場合、モデム 216 は、ブロードバンドモデムであってもよい。無線モデムも、また、通信ネットワーク 220 に対する無線接続のために用いられてもよい。トランシーバ装置 216 は、送信器 116 および受信器 132 の機能性を提供してもよく。通信チャンネル 120 は、接続 221 において具体化されてもよい。

#### 【0013】

コンピュータモジュール 201 は、通常は、少なくとも 1 つのプロセッサ 205 と、メモリユニット 206 とを含む。例えば、メモリユニット 206 は、半導体ランダムアクセスメモリ (RAM) と半導体読み取り専用メモリ (ROM) とを有してもよい。コンピュータモジュール 201 は、また、ビデオディスプレイ 214、スピーカ 217、およびマイクロホン 280 に連結されるオーディオビデオインタフェース 207 と、キーボード 202、マウス 203、スキャナ 226、カメラ 227、および必要に応じてジョイスティック、または他のヒューマンインタフェース装置 (図示せず) に連結される I/O インタフェース 213 と、外部モデム 216 およびプリンタ 215 のためのインタフェース 208 とを含むいくつかの入出力 (I/O) インタフェースを含む。いくつかの実施態様において、モデム 216 は、コンピュータモジュール 201 内に、例えばインタフェース 208 内に、組み込まれてもよい。コンピュータモジュール 201 は、また、ローカルエリアネットワーク (LAN) として知られているローカルエリア通信ネットワーク 222 に対する接続 223 を介してコンピュータシステム 200 の連結を可能にする、ローカルネットワークインタフェース 211 を有する。図 2A に図示されるように、ローカル通信ネットワーク 222 は、また、接続 224 を介してワイドネットワーク 220 に連結されてもよく、通常はいわゆる「ファイアウォール」装置または同様の機能性の装置を含むであろう。ローカルネットワークインタフェース 211 は、Ethernet (登録商標) 回路カード、Bluetooth (登録商標) 無線設備、または IEEE 802.11 無線設備を備えてもよいが、但し、多数の他のタイプのインタフェースが、インタフェース 211 のために実施されてもよい。ローカルネットワークインタフェース 211 は、また、送信器 116 および受信器 132 の機能性を提供してもよく、通信チャンネル 120 も、また、ローカル通信ネットワーク 222 において具体化されてもよい。

#### 【0014】

I/O インタフェース 208、213 は、シリアル接続およびパラレル接続のいずれか一方若しくは双方を利用してよく、前者は、通常は、ユニバーサルシリアルバス (US

10

20

30

40

50

B) 規格によって実施され、対応するUSBコネクタ(不図示)を有する。記憶装置209が設けられ、通常はハードディスクドライブ(HDD)210を含む。フロッピー(登録商標)ディスクドライブや磁気テープドライブ(不図示)などの他の記憶装置も、また、用いられてもよい。光ディスクドライブ212は、通常は、不揮発性のデータソースとして機能するために備えられる。ポータブルメモリ装置、このような光ディスク(例えばCD-ROM、DVD、ブルーレイディスク(商標))、USB-RAM、ポータブルな外部ハードドライブ、およびフロッピー(登録商標)ディスクは、例えば、コンピュータシステム200に対する適切なデータソースとして用いられてもよい。通常は、HDD210、光ドライブ212、ネットワーク220、222のいずれかは、ディスプレイ214を介した再生のために記憶された復号化ビデオデータのための、ビデオソース112として、または送信先として、動作するように構成されてもよい。

10

#### 【0015】

コンピュータモジュール201のコンポーネント205~213は、通常は、相互接続バス204を介して、および関連する当該技術分野のものに対して既知のコンピュータシステム200の従来のオペレーションモードをもたらす方式で、通信する。例えば、プロセッサ205は、接続218を用いて、システムバス204に連結される。同様に、メモリ206および光ディスクドライブ212は、接続219によってシステムバス204に連結される。記載の設備が実行され得るコンピュータの例は、IBM PCおよびその互換機、サン・マイクロシステムズのスパークステーション、アップル社のMac(商標)、または同様のコンピュータシステムを含む。

20

#### 【0016】

妥当若しくは所望の場合、ビデオエンコーダ114およびビデオデコーダ134は、下記の方法だけでなく、コンピュータシステム200を用いて実施されてもよく、ここで、ビデオエンコーダ114、ビデオデコーダ134、および後述の図10~図13の処理は、コンピュータシステム200内で実行可能な1つ以上のソフトウェアアプリケーションプログラム233として実施されてもよい。特に、ビデオエンコーダ114、ビデオデコーダ134、および記載の方法のステップは、コンピュータシステム200内で実行されるソフトウェア233内の命令231(図2Bを参照)によって達成される。ソフトウェア命令231は、1つ以上の特定のタスクを各々実行するために、1つ以上のコードモジュールとして形成されてもよい。ソフトウェアは2つの部分に分かれていてもよく、第1の部分および対応するコードモジュールは、記載の方法を実行し、第2の一部および対応するコードモジュールは、第1の部分とユーザとの間のユーザインタフェースを管理する。

30

#### 【0017】

ソフトウェアは、例えば下記の記憶装置を含むコンピュータ読み取り可能媒体内に記憶されてもよい。ソフトウェアは、コンピュータ読み取り可能媒体からコンピュータシステム200内にロードされ、その後、コンピュータシステム200によって実行される。コンピュータ読み取り可能媒体に記録されたこのようなソフトウェアまたはコンピュータプログラムを有するコンピュータ読み取り可能媒体は、コンピュータプログラムプロダクトである。コンピュータシステム200内のコンピュータプログラムプロダクトの使用は、ビデオエンコーダ114、ビデオデコーダ134、および記載の方法を実施するために有利な装置を好ましくはもたらす。

40

#### 【0018】

ソフトウェア233は、通常は、HDD210またはメモリ206内に記憶される。ソフトウェアは、コンピュータ読み取り可能媒体からコンピュータシステム200内にロードされ、コンピュータシステム200によって実行される。従って、例えば、ソフトウェア233は、光ディスクドライブ212によって読み取られる、光学的に読み取り可能なディスク記憶媒体(例えばCD-ROM)225に記憶されてもよい。

#### 【0019】

いくつかのこの事例では、アプリケーションプログラム233は、ユーザに供給され、1

50

つ以上のCD-ROM 225上で符号化され、対応する駆動装置212を介して読み取られてもよいし、または、その代わりにユーザによってネットワーク220または222から読み込まれてもよい。さらにまた、ソフトウェアは、他のコンピュータ読み取り可能なディスク記憶媒体からコンピュータシステム200内にロードすることもできる。コンピュータ読み取り可能な記憶媒体は、実行および/または処理のために、コンピュータシステム200に対して、記録された命令および/またはデータを提供する、あらゆる非一時的な有形的記憶媒体を指す。このような記憶媒体の例は、フロッピー（登録商標）ディスク、磁気テープ、CD-ROM、DVD、ブルーレイディスク、ハードディスクドライブ、ROM、集積回路、USBメモリ、光磁気ディスク、またはPCMCIAカードおよびその他同種のものなどのコンピュータ読み取り可能なカードを含み、このような装置は、  
10  
コンピュータモジュール201の内部にあるか外部にあるかを問わない。コンピュータモジュール401に対するソフトウェア、アプリケーションプログラム、命令および/またはビデオデータまたは符号化ビデオデータの提供にも関与してもよい一時的若しくは非有形的コンピュータ読み取り可能な伝送媒体の例は、別のコンピュータまたはネットワーク接続された装置に対するネットワーク接続だけでなく無線若しくは赤外線を送信チャンネル、および電子メール送信並びにウェブサイトに記録された情報やその他同種のものを含むインターネットまたはイントラネットを含む。

#### 【0020】

上述のアプリケーションプログラム233および対応するコードモジュールの第2の部分は、ディスプレイ214上に描画されるか、さもなければ表現される1つ以上のグラフィカルユーザインタフェース（GUI）を実施するために実行されてもよい。通常はキーボード202およびマウス203の操作を通じて、コンピュータシステム200およびアプリケーションのユーザは、GUIに関連するアプリケーションに対して制御コマンドを提供および/または入力する機能的に適応可能な方式で、インタフェースを操作してもよい。機能的に適応可能なユーザインタフェースの他の形式も、また、スピーカ217を介して出力される音声プロンプトおよびマイクロホン280を介して入力されるユーザ音声コマンドを利用するオーディオインタフェースなどにより、実施されてもよい。

#### 【0021】

図2Bは、プロセッサ205および「メモリ」234の詳細な概略ブロック図である。メモリ234は、図2Aのコンピュータモジュール201によってアクセスすることができる、（HDD209および半導体メモリ206を含む）すべてのメモリモジュールの論理的な集約を表現する。

#### 【0022】

コンピュータモジュール201が最初に起動される時、電源投入時自己診断テスト（POST）プログラム250が実行する。POSTプログラム250は、通常は図2Aの半導体メモリ206のROM249内に記憶される。ソフトウェアを記憶するROM249などのハードウェアデバイスは、時として、ファームウェアと呼ばれる。POSTプログラム250は、正常な機能を保証するためにコンピュータモジュール201内のハードウェアを検査し、通常は、プロセッサ205、メモリ234（209、206）、および通常はROM249内にも記憶される正常動作のための基本入出力システムソフトウェア（BIOS）モジュール251をチェックする。一旦POSTプログラム250の実行が成功すれば、BIOS251は、図2Aのハードディスクドライブ210を作動させる。ハードディスクドライブ210の起動は、ハードディスクドライブ210上に常駐するブートストラップローダプログラム252を、プロセッサ205を介して実行させる。これは、オペレーティングシステム253が動作を始めるRAMメモリ206内に、オペレーティングシステム253をロードする。オペレーティングシステム253はプロセッサ管理、メモリ管理、装置管理、記憶管理、ソフトウェアアプリケーションインタフェース、および総括的なユーザインタフェースを含む様々な高レベルの機能を実現する、プロセッサ205により実行可能な、システムレベルアプリケーションである。

#### 【0023】

10

20

30

40

50



オペレーティングシステム 253 は、コンピュータモジュール 201 上で動作する各処理またはアプリケーションが、別の処理に対して割り当てられたメモリと衝突せずに実行するのに十分なメモリを有することを保証するためにメモリ 234 (209、206) を管理する。さらに、図 2 A のコンピュータシステム 200 において利用可能な異なるタイプのメモリは、各処理が効率的に動作することができるように、適切に用いられなければならない。従って、集約されたメモリ 234 は、(特に断らない限り)メモリの特定のセグメントがどのように割り当てられるか示すのではなく、むしろ、コンピュータシステム 200 によってアクセス可能なメモリおよびこのようなものがどのように用いられるかの一般的概念を提供するように意図される。

#### 【0024】

図 2 B において示されるように、プロセッサ 205 は、制御部 239 と、算術論理演算ユニット (ALU) 240 と、時としてキャッシュメモリと呼ばれるローカルまたは内部メモリ 248 とを含む、いくつかの機能モジュールを含む。キャッシュメモリ 248 は、通常は、レジスタ部内にいくつかの記憶レジスタ 244 ~ 246 を含む。1 つ以上の内部バス 241 は、これらの機能モジュールを機能的に相互に連結する。また、プロセッサ 205 は、通常は、接続 218 を用いて、システムバス 204 を介して外部装置と通信するための 1 つ以上のインタフェース 242 を有する。メモリ 234 は、接続 219 を用いて、バス 204 に連結される。

#### 【0025】

アプリケーションプログラム 233 は、条件分岐およびループ命令を含んでもよい命令 231 のシーケンスを含む。プログラム 233 は、また、プログラム 233 の実行に用いられるデータ 232 を含んでもよい。命令 231 およびデータ 232 は、それぞれ、メモリ位置 228、229、230 および 235、236、237 に記憶される。命令 231 の相対的なサイズおよびメモリ位置 228 ~ 230 に応じて、特定の命令は、メモリ位置 230 内に示される命令によって表現されるような単一のメモリ位置内に記憶されてもよい。交互に、メモリ位置 228、229 に示される命令セグメントによって表現されるように、命令は、各々が個別のメモリ位置に記憶されるいくつかの部分にセグメント化されてもよい。

#### 【0026】

一般的に、プロセッサ 205 は、プロセッサ 205 の中で実行される 1 セットの命令を与えられる。プロセッサ 205 は、別の命令のセットを実行することによってプロセッサ 205 が反応する、次の入力を待つ。各入力は、すべて図 2 A に表現されている、入力装置 202、203 の 1 つ以上によって生成されたデータ、ネットワーク 220、202 の 1 つを介して外部ソースから受信されたデータ、記憶装置 206、209 の 1 つから検索されたデータ、または対応する読み取り装置 212 内に挿入された記憶媒体 225 から検索されたデータを含む、いくつかのソースの 1 つ以上から提供されてもよい。1 セットの命令の実行は、いくつかのケースにおいてデータの出力をもたらしてもよい。実行は、また、データまたは変数をメモリ 234 に記憶することを含んでもよい。

#### 【0027】

ビデオエンコーダ 114、ビデオデコーダ 134 および記載の方法は、対応するメモリ位置 255、256、257 のメモリ 234 内の記憶される入力変数 254 を用いてもよい。ビデオエンコーダ 114、ビデオデコーダ 134 および記載の方法は、対応するメモリ位置 262、263、264 のメモリ 234 内に記憶される出力変数 261 を生成する。中間変数 258 は、メモリ位置 259、260、266、267 に記憶されてもよい。

#### 【0028】

図 2 B のプロセッサ 205 を参照すると、レジスタ 244、245、246、算術論理演算ユニット (ALU) 240、および制御部 239 は、プログラム 233 を作成する命令セット内の命令ごとの「フェッチ、復号化、実行」サイクルを実行するために必要とされるマイクロオペレーションのシーケンスを実行するために相互に動作する。各々のフ

10

20

30

40

50

エッチ、復号化、および実行のサイクルは、以下の動作を備える。

【 0 0 2 9 】

( a ) メモリ位置 2 2 8、2 2 9、2 3 0 から命令 2 3 1 をフェッチまたは読み込むフェッチ動作。

( b ) どの命令がフェッチされたかを制御部 2 3 9 が判定する復号化動作。

( c ) 制御部 2 3 9 および / または A L U 2 4 0 が命令を実行する実行動作。

【 0 0 3 0 】

その後、次の命令のためのさらなるフェッチ、復号化、および実行のサイクルが実行されてもよい。同様に、制御部 2 3 9 がメモリ位置 2 3 2 に対する値を記憶するまたは書き込む記憶サイクルが実行されてもよい。

10

【 0 0 3 1 】

後述する図 1 0 ~ 図 1 3 の処理の各ステップまたはサブ工程は、1 つ以上のプログラム 2 3 3 のセグメントに関連づけられ、通常は、プログラム 2 3 3 の言及されたセグメントのために命令セット内の命令ごとにフェッチ、復号化、および実行サイクルを実行するために共動するプロセッサ 2 0 5 内のレジスタ部 2 4 4、2 4 5、2 4 7、A L U 2 4 0、および制御部 2 3 9 によって実行される。

【 0 0 3 2 】

図 3 は、ビデオエンコーダ 1 1 4 の機能モジュールを示す概略ブロック図である。図 4 は、ビデオデコーダ 1 3 4 の機能モジュールを示す概略ブロック図である。ビデオエンコーダ 1 1 4 およびビデオデコーダ 1 3 4 は、図 2 A および図 2 B に図示するように、汎用コンピュータシステム 2 0 0 を用いて実施されてもよく、ここではコンピュータシステム 2 0 0 内の専用ハードウェアによって実施されてもよいし、または、コンピュータシステム 2 0 0 内で実行可能なソフトウェア、例えば、ハードディスクドライブ 2 0 5 上に常駐し、且つプロセッサ 2 0 5 によって実行を制御されるソフトウェアアプリケーションプログラム 2 3 3 の 1 つ以上のソフトウェアコードモジュールなどの様々な機能モジュールによって実施されてもよいし、または、コンピュータシステム 2 0 0 内で実行可能な専用ハードウェアおよびソフトウェアの組み合わせによって実施されてもよい。ビデオエンコーダ 1 1 4、ビデオデコーダ 1 3 4 および記載の方法は、代わりに、記載の方法の機能またはサブ機能を実行する 1 つ以上の集積回路などの専用ハードウェアにより実現されてもよい。このような専用ハードウェアは、グラフィックプロセッサ、デジタル信号プロセッサ、特定用途向け集積回路 ( A S I C )、フィールドプログラマブルゲートアレイ ( F P G A ) または 1 つ以上のマイクロプロセッサおよび関連メモリを含む。特に、ビデオエンコーダ 1 1 4 は、モジュール 3 2 0 ~ 3 4 4 を備え、ビデオデコーダ 1 3 4 は、ソフトウェアアプリケーションプログラム 2 3 3 の 1 つ以上のソフトウェアコードモジュールとして各々実装されてもよいモジュール 4 2 0 ~ 4 3 4 を備える。

20

30

【 0 0 3 3 】

図 3 のビデオエンコーダ 1 1 4 は高効率動画符号化方式 ( H E V C ) ビデオ符号化パイプラインの例であるが、モジュール 3 2 0 - 3 4 4 によって実行される処理段階は、V C - 1 または H . 2 6 4 / M P E G - 4 A V C などの他のビデオコーデックと共通である。ビデオエンコーダ 1 1 4 は、一連のフレームとして、取得されたフレームデータなどの取得されたフレームデータを受信し、各フレームは 1 つ以上の色チャンネルを含む。各フレームは、色チャンネルごとに 1 つのサンプルグリッドを備える。色情報は、勧告 I T U - R B T . 7 0 9 ( 「 Y U V 」 ) などの「色空間」を用いて表現されるが、他の色空間も可能である。Y U V 色空間が用いられる場合、色チャンネルは、輝度チャンネル ( 「 Y 」 ) および 2 つの色差チャンネル ( 「 U 」 および 「 V 」 ) を含む。さらに、異なる情報量は、画像のサンプリングに応じて、または取得されたフレームデータを再サンプリングするフィルタリングアプリケーションを通じて、各色チャンネルのサンプルグリッド内に含まれてもよい。「色差フォーマット」として知られているいくつかのサンプリングアプローチが存在し、そのうちのいくつかは、図 5 A および図 5 B を参照して説明する。

40

【 0 0 3 4 】

50

ビデオエンコーダ114は、フレームデータ310などの取得されたフレームデータの各フレームを、一般的に「符号化ツリーブロック」(CTB)と呼ばれる領域に分割する。各符号化ツリーブロック(CTB)は、「符号化単位」(CU)の集合への一部のフレームの階層的四分木細分割(hierarchical quad-tree subdivision)を含む。符号化ツリーブロック(CTB)は、一般的に、64×64輝度サンプルの領域を占めるが、16×16または32×32などの他のサイズも可能である。いくつかのケースにおいて、さらに大規模なサイズ(例えば128×128)が用いられてもよい。符号化ツリーブロック(CTB)は、新規の階層レベルを生成するために、4つの等しいサイズの領域への分割をへて、細分割されてもよい。分割は、再帰的に適用されても良く、その結果、四分木階層内をもたらす。符号化ツリーブロック(CTB)側の寸法が常に2の累乗であるので、四分木分割は、常に幅および高さの半値化するので、領域側寸法も、また、常に2の累乗である。それ以上分割領域が行われない場合、「符号化単位」(CU)は、領域内に存在すると言える。符号化ツリーブロックのトップレベルで分割が実行されない場合、符号化ツリーブロック全体を占有する領域は「最大符号化単位」(LCU)と一般的に呼ばれる1つの符号化単位(CU)を含む。最小サイズは、また、8×8輝度サンプルによって占められた領域などのように、各々の符号化単位ごとに存在するが、他の最小サイズも可能である。このサイズの符号化単位は、一般的に「最小符号化単位」(SCU)と呼ばれる。この四分木階層の結果として、符号化ツリーブロック(CTB)の全体は、1つ以上の符号化単位(CU)によって占有される。

ビデオエンコーダ114は、各符号化単位(CU)に対して、一般的に「予測単位」(PU)と呼ばれるサンプルの1つ以上の配列を生成する。各符号化単位(CU)における予測単位(PU)の様々な配列は、予測単位(PU)がオーバーラップせず、符号化単位(CU)の全体が1つ以上の予測単位(PU)によって占有されるという要件によって可能である。このスキームは、予測単位(PU)がフレーム領域全体をカバーすることを保証する。

#### 【0035】

ビデオエンコーダ114は、マルチプレクサモジュール340、予測単位(PU)382からの出力によって動作する。差分モジュール344は、フレームデータ310の符号化ツリーブロック(CTB)の符号化単位(CU)から、予測単位(PU)382とデータサンプルの対応する2次元配列との間の差分を出力し、差分は「残差サンプル配列」360として知られている。差分モジュール344からの残差サンプル配列360は、変換モジュール320によって受信されるが、変換モジュール320は「順方向変換」を適用することによって残差サンプル配列360を空間表現から周波数領域表現に変換する(または「符号化」する)。変換モジュール320は、一般的に「変換ツリー」と呼ばれる1つ以上の変換単位(TU)への符号化単位(CU)の階層的細分割において変換単位(TU)における各変換のための変換係数362を生成する。開発中の高効率動画符号化方式(HEVC)規格に関しては、周波数領域表現への変換は、修正離散コサイン変換(DCT)を用いて実行され、修正離散コサイン変換では、従来のDCTは、シフトおよび加算を用いて実施されるように修正される。残差サンプル配列360および変換係数362のための様々なサイズは、サポートされた変換サイズに従って可能である。開発中の高効率動画符号化方式(HEVC)規格において、変換は、32×32、16×16、8×8、および4×4などの特定のサイズを有する2次元配列のサンプルに対して実行される。ビデオエンコーダ114に対して利用可能な変換サイズの所定のセットは、したがって、存在すると言える。さらに、以上に予示されるように、変換サイズのセットは、輝度チャンネルと色差チャンネルとの間で異なってもよい。2次元変換は一般的に「分離可能に」構成され、2次元配列のサンプル上の一方向(例えば行上)で動作する1次元変換の第1のセットとして、その後、他の方向(例えば列上)の1次元変換の第1のセットから出力される2次元配列のサンプル上で動作する1次元変換の第2のセットとして、実施を可能とする。同一の幅と高さを有する変換は、一般的に「正方形変換」と呼ばれる。さらに、異なる幅と高さを有する変換も、また可能であり、一般的に「非正方形変換」と呼ばれる

10

20

30

40

50

。最適化された変換の実施態様は、 $4 \times 4$  変換モジュールまたは  $8 \times 8$  変換モジュールなどの特定のハードウェアまたはソフトウェアモジュールへの行および列の 1 次元変換を組み合わせてもよい。たとえそれらが頻繁に用いられなくても、より大きな寸法を有する変換は、実施するために多くの量の回路を要する。従って、 $32 \times 32$  の最大変換サイズが、開発中の高効率動画像符号化方式 (HEVC) 規格内に存在する。変換の実施の統合的な本質は、また、通常は全体に新規のハードウェアが実施されることを要するので、対応する正方形変換から存在する既存の 1 次元変換ロジックを再使用する代わりに、サポートされた非正方形変換サイズの数 を低減するための優先度を導入する。変換は、輝度チャンネルおよび色差チャンネルの両方に適用される。変換単位 (TU) に関する輝度チャンネルと色差チャンネルとの取扱の差異が存在し、差異に関しては図 5 A および図 5 B を参照して以下に論じる。各変換ツリーは、1 つの符号化単位 (CU) を占有し、変換ツリー (四分木) 階層の各リーフノードにおいて 1 つの変換単位 (TU)、各変換単位 (TU) は、サポートされた変換サイズの変換を利用することができる、を含む階層への符号化単位 (CU) の四分木分解として定義される。符号化ツリーブロック (CTB) と同様に、符号化単位 (CU) の全体は 1 つ以上の変換単位 (TU) によって占有されることが必要である。変換ツリー四分木階層の各レベルにおいて、それ以上分割が存在しない場合は現在の階層レベルにおいて、「符号化ブロックフラグ値」は、各色チャンネル内での変換の存在が有りうることを信号伝達し、または下位階層レベルが結果として生じる変換単位 (TU) への少なくとも 1 つの変換を含んでもよいと信号伝達する。符号化ブロックフラグ値がゼロである場合、現在の階層レベルまたはより低い階層レベルにおける、変換ツリーのいかなる変換単位 (TU) の対応する色チャンネルに対しても、変換は実行されない。符号化ブロックフラグ値が 1 である場合、領域は、少なくとも 1 つのゼロでない残差係数を有しなければならない変換を含む。このように、各色チャンネルに対して、ゼロまたはそれ以上の変換は、符号化単位 (CU) なしから符号化単位 (CU) 全体まで変動する符号化単位 (CU) の領域の一部をカバーしてもよい。個別の符号化ブロックフラグ値は、各色チャンネルに存在する。1 つの有りうる符号化ブロックフラグ値のみがあるケースが存在する場合、各符号化ブロックフラグ値は、符号化される必要がない。

#### 【0036】

その後、変換係数  $362$  は、スケール/量子化モジュール  $322$  に入力され、残差係数配列  $364$  を生成するために決定された量子化パラメータ  $384$  によって、スケールされ、量子化される。スケール/量子化処理は、決定された量子化パラメータ  $384$  値に依存して、精度の損失をもたらす。決定された量子化パラメータ  $384$  の高い値は、変換係数から大きな情報が失われる結果をもたらす。これにより、ビデオデコーダ  $134$  からの出力の視覚的品質を低減することを犠牲にして、ビデオエンコーダ  $114$  によって達成された圧縮を増大する。決定された量子化パラメータ  $384$  はフレームデータ  $310$  の各フレームの符号化の間に適応されてもよいし、または、全体のフレームなどフレームデータ  $310$  の一部に固定されてもよい。また、決定された量子化パラメータ  $384$  の他の適応は、個別の値による異なる残差係数の量子化などで、可能である。残差係数配列  $364$  および決定された量子化パラメータ  $384$  は、逆スケールモジュール  $326$  に対する入力として用いられるが、逆スケールモジュール  $326$  は、残差係数配列  $364$  のリスケールされたバージョンであるリスケールされた変換係数配列  $366$  を生成するためにスケール/量子化モジュール  $322$  によって実行されたスケール処理を反転させる。

#### 【0037】

残差係数配列  $364$  および決定された量子化パラメータ  $384$  も、符号化ビットストリーム  $312$  (または「ビデオビットストリーム」) 内の残差係数を符号化するエンタープライズエンコーダモジュール  $324$  に対する入力として使用される。各変換単位 (TU) 内の各変換の残差係数配列  $364$  は、「サブブロック」として一般的に知られているグループにおいて符号化される。これがサブブロック処理に関連するロジックの再使用を可能にするので、サブブロックは、好ましくは、変換のサイズに関わらず、同一の寸法を有すべきである。1 つのサブブロック内の残差係数は、一般的に「係数グループ」と呼ばれ、各

10

20

30

40

50

係数グループに対して、係数グループフラグは、一般的に、係数グループ内の少なくとも1つの残差係数がゼロでないか否かを示すために符号化される。いくつかのケースにおいて、係数グループフラグは、暗示され、したがって符号化されない。フラグは、残差係数がゼロでない（「有意」）か、ゼロである（「非有意」）か、を示すために1である係数グループフラグ値を有する係数グループに対する属する各残差係数に対して符号化される。スケール/量子化モジュール322から起因する精度の損失のために、スケール処理された変換係数配列366は、オリジナルの変換係数362と同一でない。逆スケールモジュール326からスケール処理された変換係数配列366は、その後、逆変換モジュール328に出力される。ビデオデコーダ134で生成される空間領域表現と同一のスケール変更された変換係数配列366の空間領域表現368を生成するために、逆変換モジュール328は、周波数領域から空間領域に対して逆変換を実行する。

10

#### 【0038】

動き推定モジュール338は、一般的にはメモリ206内に構成されているフレームバッファモジュール332内に記憶された1セット以上のフレームからの前フレームデータと、フレームデータ310とを比較することによって、運動ベクトル374を生成する。フレームのセットは「基準画像リスト」として知られている。運動ベクトル374は、その後、運動ベクトル374から派生した空間オフセットを考慮に入れて、フレームバッファモジュール332内に記憶されたサンプルをフィルタリングすることによって、インター予測された予測単位(PU)376を生成する動き補償モジュール334に入力される。図3には図示されないが、運動ベクトル374も、また、符号化ビットストリーム312内で符号化するためにエントロピーエンコーダモジュール324に対する構文要素として渡される。イントラフレーム予測モジュール336は、総和モジュール342から取得されたサンプル370を用いて、インター予測された予測単位(PU)378を生成する。総和モジュール342は、マルチプレクサモジュール340からの予測単位(PU)382と、逆変換モジュール328からの空間領域表現368とを総和する。イントラフレーム予測モジュール336は、また、符号化ビットストリーム312に符号化するためのエントロピーエンコーダ324に対して送信されるイントラ予測モード380を生成する。

20

#### 【0039】

予測単位(PU)は、イントラ予測方法またはインター予測方法のいずれかを用いて生成されてもよい。イントラ予測方法は、予測単位(PU)内の基準サンプルを生成するために(通常は、予測単位の上および左に)予め復号化された予測単位(PU)に隣接するサンプルを使用する。「イントラ予測モード」と呼ばれる、イントラ予測を様々な方向にすることは可能である。インター予測方法は、選択された基準フレームからブロックまで参照するために動きベクトルを利用する。ブロックがサブサンプル精度(例えばサンプルの8分の1)までのあらゆるアラインメントを有するので、フィルタリングは、予測単位(PU)のための基準サンプルのブロックを生成するのに必要となる。結果として生じる符号化ビットストリーム312の所望のビットレートと、イントラ予測またはインター予測方法のいずれかによって導入された画質歪みの量との間のレート歪みトレードオフに従って、用いる方法が決定される。イントラ予測が用いられる場合、1つのイントラ予測モードは、また、レート歪みトレードオフに従ってイントラ予測の可能なモードのセットから選択される。マルチプレクサモジュール340は、レート歪みアルゴリズムによって行なわれた決定に依存して、イントラフレーム予測モジュール336からインター予測された基準サンプル378を選択するか、または動き補償ブロック334からインター予測された予測単位(PU)376を選択する。総和モジュール342は、デブロッキングフィルタモジュール330に入力される総和370を生成する。デブロッキングフィルタモジュール330は、メモリ206内で構成されたフレームバッファモジュール332に対して書き込まれる、デブロッキングされたサンプル372を生成して、ブロック境界に沿ってフィルタリングを実行する。フレームバッファモジュール332は、基準画像リストの一部として将来の参考のための1つ以上の過去フレームからのデータを保持するのに十分

30

40

50

な容量を有するバッファである。

【 0 0 4 0 】

開発中の高効率動画画像符号化方式（H E V C）規格に関しては、エントロピーエンコーダ 3 2 4 によって生成された符号化ビットストリーム 3 1 2 は、ネットワーク抽象化層（N A L）単位に描写される。一般的に、フレームの各スライスは、1つのN A L単位内に含まれる。エントロピーエンコーダ 3 2 4 は、コンテキスト適応可能な2進算術演算符号化（C A B A C）アルゴリズムを実行することによって符号化ビットストリーム 3 1 2 内に残差係数配列 3 6 4、イントラ予測モード 3 8 0、動きベクトル、および他のパラメータ（まとめて「構文要素」と呼ばれる）を符号化する。構文要素は、「構文構造」にグループ化され、これらのグループ化は、階層構造を記載するために再帰を含んでもよい。イントラ予測モードなどの序数値、または動きベクトルなどの整数値に加えて、構文要素は、四分木分割などを示すために、フラグを含む。動き推定モジュール 3 3 8 および動き補償モジュール 3 3 4 は、フレームデータ 3 1 0 内のフレーム間の動きの高精度モデリングを可能にする、輝度サンプルの 1 / 8 の精度を有する、運動ベクトル 3 7 4 上で動作する。

10

【 0 0 4 1 】

図 4 のビデオデコーダ 1 3 4 は、高効率動画画像符号化方式（H E V C）ビデオ復号化パイプラインを参照して説明するが、モジュール 4 2 0 ~ 4 3 4 によって実行される処理段階は、エントロピーコーディング（H . 2 6 4 / M P E G - 4 A V C、M P E G 2 および V C - 1）を用いる他のビデオコーデックと共通である。符号化ビデオ情報は、また、メモリ 2 0 6、ハードディスクドライブ 2 1 0、C D - R O M、ブルーレイ（商標）ディスク、または他のコンピュータ読み取り可能記録媒体から読み取ってもよい。または、符号化ビデオ情報は、通信ネットワーク 2 2 0 に接続されたサーバまたは無線周波数受信器などの外部ソースから受信してもよい。

20

【 0 0 4 2 】

図 4 でわかるように、符号化ビットストリーム 3 1 2 などの受信されたビデオデータは、ビデオデコーダ 1 3 4 に入力される。符号化ビットストリーム 3 1 2 は、メモリ 2 0 6、ハードディスクドライブ 2 1 0、C D - R O M、ブルーレイ（商標）ディスク、または他のコンピュータ読み取り可能記録媒体から読み取ってもよい。または、符号化ビットストリーム 3 1 2 は、通信ネットワーク 2 2 0 に接続されたサーバまたは無線周波数受信器などの外部ソースから受信してもよい。符号化ビットストリーム 3 1 2 は、復号化される取得されたフレームデータを表現する、符号化された構文要素を含む。

30

【 0 0 4 3 】

符号化ビットストリーム 3 1 2 は、符号化ビットストリーム 3 1 2 から構文要素を抽出し、且つ構文要素の値をビデオデコーダ 1 3 4 内の他のブロックに渡す、エントロピーデコーダモジュール 4 2 0 に入力される。エントロピーデコーダモジュール 4 2 0 は、符号化ビットストリーム 3 1 2 から構文要素を復号化するためにコンテキスト適応可能な2進算術演算符号化（C A B A C）アルゴリズムを適用する。復号化された構文要素は、ビデオデコーダ 1 3 4 内のパラメータを再製するために用いられる。パラメータは、0以上の残差係数配列 4 5 0、動きベクトル 4 5 2、および予測モード 4 5 4 を含む。残差係数配列 4 5 0 は、逆スケール/変換モジュール 4 2 2 に渡され、動きベクトル 4 5 2 は、動き補償モジュール 4 3 4 に渡され、予測モード 4 5 4 は、イントラフレーム予測モジュール 4 2 6 およびマルチプレクサ 4 2 8 に渡される。逆スケール/変換モジュール 4 2 2 は、再製された変換係数を生成するために残差係数データに逆スケール処理を実行する。逆スケール/変換モジュール 4 2 2 は、その後、再製された変換係数を周波数領域表現から空間領域表現に変換する（「または復号化する」）ために「逆変換」を適用して、残差サンプル配列 4 5 6 を生成する。逆スケール/変換モジュール 4 2 2 における逆変換は、逆変換 3 2 8 と同一の動作を実行する。そのため、逆スケール/変換モジュール 4 2 2 は、開発中の高効率動画画像符号化方式（H E V C）規格に準拠した符号化ビットストリーム 3 1 2 を復号化するのに必要な変換サイズの所定のセットを提供するように、構成されなけれ

40

50

ばならない。

【 0 0 4 4 】

動き補償モジュール 4 3 4 は、出力復号化フレームデータの予測である、予測単位 ( P U ) に対してインター予測された予測単位 ( P U ) 4 6 2 を生成するためにメモリ 2 0 6 内で構成された、フレームバッファブロック 4 3 2 からの基準フレームデータ 4 6 0 に組み合わされた、エントロピーデコーダモジュール 4 2 0 からの動きベクトル 4 5 2 を用いる。現在の予測単位がイントラ予測を用いて符号化されたことを予測モード 4 5 4 が示す場合、イントラフレーム予測モジュール 4 2 6 は、予測モード 4 5 4 によって供給された予測単位 ( P U ) および予測方向に隣接するサンプルを空間的に用いて予測単位 ( P U ) に対してインター予測された予測単位 ( P U ) 4 6 4 を生成する。空間的に近隣のサンプルは、総和モジュール 4 2 4 から出力された総和 4 5 8 から取得される。マルチプレクサモジュール 4 2 8 は、現在の予測モード 4 5 4 に基づいて、予測単位 ( P U ) 4 6 6 に対してインター予測された予測単位 ( P U ) 4 6 4 またはインター予測された予測単位 ( P U ) 4 6 2 を選択する。マルチプレクサモジュール 4 2 8 から出力された予測単位 ( P U ) 4 6 6 は、その後、デブロッキングフィルタモジュール 4 3 0 の各々に対して入力される総和 4 5 8 を生成する総和モジュール 4 2 4 およびイントラフレーム予測モジュール 4 2 6 によって、逆スケール / 変換モジュール 4 2 2 から残差サンプル配列 4 5 6 に加えられる。デブロッキングフィルタモジュール 4 3 0 は、可視のアーチファクトを平滑化するために、変換単位 ( T U ) 境界などのデータブロック境界に沿ってフィルタリングを実行する。デブロッキングフィルタモジュール 4 3 0 の出力は、メモリ 2 0 6 内で構成されたフレームバッファモジュール 4 3 2 に書き込まれる。フレームバッファモジュール 4 3 2 は、将来の参考のための 1 つ以上の復号化フレームを保持するのに十分なストレージを提供する。復号化フレーム 4 1 2 も、また、フレームバッファモジュール 4 3 2 からディスプレイ装置 1 3 6 などのディスプレイ装置まで出力される。

【 0 0 4 5 】

図 5 A および図 5 B は、各々、それぞれ 4 : 2 : 0 および 4 : 2 : 2 色差フォーマットを用いて符号化されたフレーム部分 5 0 0 およびフレーム部分 5 1 0 のサンプルグリッドを示す。ビデオエンコーダ 1 1 4 に対して構成パラメータとして色差フォーマットが指定され、ビデオエンコーダ 1 1 4 は、色差フォーマットを指定する符号化ビットストリーム 3 1 2 に「 chroma\_\_format\_\_idc 」構文要素を符号化する。ビデオデコーダ 1 3 4 は、使用中の色差フォーマットを決定するために符号化ビットストリーム 3 1 2 から「 chroma\_\_format\_\_idc 」構文要素を復号化する。例えば、4 : 2 : 0 色差フォーマットが使われている場合、 chroma\_\_format\_\_idc の値は 1 であり、4 : 2 : 2 色差フォーマットが使用中である場合、 chroma\_\_format\_\_idc の値は 2 であり、4 : 4 : 4 色差フォーマットが使用中である場合、 chroma\_\_format\_\_idc の値は 3 である。図 5 A および図 5 B において、輝度サンプル位置 5 0 1 などの輝度サンプル位置は「 X 」のシンボルを用いて図示され、色差サンプル位置 5 0 2 などの色差サンプル位置は「 O 」のシンボルを用いて図示される。示されたポイントでのフレーム部分 5 0 0 をサンプリングすることによって、4 : 2 : 0 色差フォーマットが適用される場合、サンプルグリッドは、各色チャンネルに対して取得される。各輝度サンプル位置 X において、輝度チャンネル ( 「 Y 」 ) がサンプリングされ、各々の色差サンプル位置 O において、双方の色差チャンネル ( 「 U 」 および 「 V 」 ) がサンプリングされる。図 5 A に示されるように、各色差サンプル位置に対して輝度サンプル位置の 2 × 2 配置が存在する。フレーム部分 5 1 0 内に示された輝度サンプル位置での輝度サンプルおよび色差サンプル位置での色差サンプルをサンプリングすることによって、サンプルグリッドは、4 : 2 : 2 色差フォーマットが適用される場合に、各色チャンネルに対して取得される。色チャンネルに対する同一のサンプル割り当ては、フレーム部分 5 1 0 に対して、フレーム部分 5 0 0 と同様に作成される。フレーム部分 5 0 0 とは対照的に、2 倍の数の色差サンプル位置がフレーム部分 5 1 0 内に存在する。フレーム部分 5 1 0 において、色差サンプル位置は、2 番目の輝度サンプル位置ごとに配置される。従って、図 5 B において、

各色差サンプル位置に対して、 $2 \times 1$ 輝度サンプル位置の配置が存在する。

【0046】

以上、変換単位の様々な許容寸法を、輝度サンプルの単位において、説明した。輝度チャンネルに対して適用された変換によってカバーされる領域は、したがって、変換単位寸法と同一の寸法を有することになる。変換単位も色差チャンネルを符号化するので、各々の色差チャンネルに対して適用される変換は、使用中の特定の色差フォーマットに従って適応された寸法を有することになる。例えば、 $4:2:0$ 色差フォーマットが使用中である場合、 $16 \times 16$ 変換単位(TU)は、 $16 \times 16$ 変換を輝度チャンネルに対して用い、 $8 \times 8$ 変換を各色差チャンネルに対して用いることになる。 $4 \times 4$ 変換が輝度チャンネルに対して用いられる1つの特殊なケースは、色差チャンネルに対して用いることができた( $4:2:0$ 色差フォーマットが適用される場合に)利用可能な対応する $2 \times 2$ 変換、または( $4:2:2$ 色差フォーマットが適用される場合に)利用可能な $4 \times 2$ 変換が存在しないという場合である。この特殊なケースにおいては、各色差チャンネルに対する $4 \times 4$ 変換は、複数の輝度変換によって占有された領域をカバーしてもよい。

10

【0047】

図6Aは、フレームの符号化ツリーブロック(CTB)600内の(太い境界線で表現された)符号化単位(CU)602の例示的な変換ツリーの概略図である。単一の四分木細分割は、符号化ツリーブロック(CTB)600を、符号化単位(CU)602などの4つの $32 \times 32$ 符号化単位(CU)に分割する。例示的な変換ツリーは、符号化単位(CU)602内に存在する。例示的な変換ツリーは、図6A(例えば変換単位#9(TU)604)のように番号付けされた10個の変換単位(TU)をもたらず、いくつかの四分木細分割を含む。変換単位#1~#10は、符号化単位(CU)602の全体をカバーする。各四分木細分割は、領域を4つの象限に空間的に分割し、4つのさらに小さな領域をもたらず。各変換単位(TU)は、変換ツリー内の変換単位(TU)の階層レベルに対応する変換深度値を有する。階層レベルは、四分木細分割の数が、終了した四分木細分割の前に実行され、結果として、対応する領域を占有する変換単位(TU)のインスタンスをもたらずことを示す。例えば変換単位#9(TU)604は、符号化単位(CU)602の4分の1の領域を占めており、そのため、1である変換深度を有する。各変換単位(TU)は、輝度サンプルグリッド上に変換単位(TU)を含む領域の寸法として一般的に記載される、関係付けられたサイズ(または「変換サイズ」)を有する。サイズは、符号化単位(CU)のサイズおよび変換深度に依存する。ゼロである変換深度を有する変換単位(TU)は、対応する符号化単位(CU)のサイズと同等サイズを有する。各変換深度のインクリメントは、所定の変換深度での変換ツリー内にある変換単位(TU)のサイズの半値をもたらず。フレームが一つの輝度チャンネルおよび複数の色差チャンネルを含むので、符号化単位(CU)602は、輝度サンプルグリッドおよび色差サンプルグリッドの両方の領域を占有し、したがって、各変換単位(TU)は、輝度サンプルグリッド上の輝度サンプルおよび色差サンプルグリッド上の色差サンプルの両方を記載する情報を含む。各変換単位(TU)に対する情報の性質は、ビデオエンコーダ114またはビデオデコーダ134の処理段階に依存する。変換モジュール320への入力および逆スケール/変換モジュール422の出力時、残差サンプル配列360、456は、それぞれ、空間領域内の各変換単位(TU)に対する情報を含んでいる。残差サンプル配列360、456は、輝度チャンネルと色差チャンネルとの間の処理時の差分に起因して、「色差残差サンプル配列」および「輝度残差サンプル配列」にさらに分割されてもよい。スケール/量子化モジュール322からの出力および逆スケール/変換モジュール422への入力時において、残差係数配列364、450は、それぞれ、周波数領域内の各変換単位(TU)に対する情報を含む。残差係数配列364、450は、輝度チャンネルと色差チャンネルとの間の処理の差分に起因して、「色差残差係数配列」および「輝度残差係数配列」にさらに分割されてもよい。

20

30

40

【0048】

図6Bは、輝度サンプルグリッド上の $32 \times 32$ 輝度サンプル配列を占有する、1セッ

50



トの変換単位 (TU) を含んで符号化単位 (CU) 602 を占有する、 $32 \times 32$  符号化単位 (CU) の輝度チャンネルに関して、図 6 A の例示的な変換ツリーに対応する、例示的な変換ツリー 630 を図示する。図 7 は、例示的な変換ツリー 630 を表現するデータ構造 700 を図示する。図 6 B において、1 ~ 10 と番号付けられたボックスは、(いくつかの変換単位 (TU) 640 によって例示された) 領域 632 内に存在する変換単位を示し、各ボックスは、さらに細分割された (破線の境界線を有するボックスによって示された) 領域内に含まれる。

【0049】

図 6 B において、1 および 9 と番号付けられたボックスは、輝度チャンネルに対する  $16 \times 16$  変換を含み、2、3 および 8 と番号付けられたボックスは、輝度チャンネルに対する  $8 \times 8$  変換を含む、4 ~ 7 と番号付けられたボックスは、輝度チャンネルに対する  $4 \times 4$  変換を含む。これらのボックスの各々に対応する領域 (破線のボックス) は、変換の存在を示すために符号化ブロックフラグ値 1 を有する。

10

【0050】

各色チャンネルに対する変換の有無は、以下に論じられるように、ビットストリームの各々の符号化 / 復号化に用いられるが、ビットストリームにおいては送信される必要はない、個別の符号化ブロックフラグ値によって指定される。結果的に、エントロピーデコーダ 420 から出力される残差係数配列 450 の数は、符号化ブロックフラグ値に依存する。どの色チャンネル内にも有意係数がない場合、エントロピーデコーダ 420 から出力される残差係数配列 450 の数は、ゼロである。

20

【0051】

図 7 において、円は、対応する円の内部に示される分割変換フラグ値とともに分割変換フラグ値を表現する。図 7 において、三角形は、対応する三角形の内部に示される符号化ブロックフラグ値とともに符号化ブロックフラグ値を表現する。正方形は、図 6 B にある変換番号に一致するように番号付けされた各変換とともに変換単位を表現する。

【0052】

例示的な変換ツリー 630 の最上部の階層レベルは、 $32 \times 32$  符号化単位 (CU) を占有する領域 632 を含んでいる。分割変換フラグ値 702 は、領域 632 が 4 つの  $16 \times 16$  領域 (例えば領域 634) に細分割されることを示し、したがって例示的な変換ツリー 630 の「非リーフ」ノードを定義する。各  $16 \times 16$  領域に関しては、分割変換フラグ値 704 などのさらなる分割変換フラグ値は、それぞれの  $16 \times 16$  領域が 4 つの  $8 \times 8$  領域にさらに細分割されるべきであることを示す。例えば、ゼロである分割変換フラグ値 704 によって示されるように、領域 634 はさらに細分割されず、したがって例示的な変換ツリー 630 の「リーフ」ノードを定義する。対照的に、1 である分割変換フラグ値 712 によって示されるように、領域 638 は、領域 636 などの 4 つの  $4 \times 4$  領域にさらに細分割される。変換ツリー 630 内に存在する再帰的な分割構造は、符号化ツリーブロック (CTB) 内にある四分木分割に類似する。輝度チャンネルに関しては、四分木の「リーフ」ノードにおいて、変換単位 (TU) 内の変換の有無が符号化ブロックフラグ値によって信号伝達され、例えば、1 である符号化ブロックフラグ値 708 は、領域 634 内の変換 710 の有無を示す。

30

40

【0053】

変換が各領域内の残差データを表現するために用いられてもよいので、輝度チャンネルに対して  $4 \times 4$  輝度サンプルなど、領域がサポートされた最小変換サイズより小さくなることを許可しない。その上、利用可能な最大変換サイズより大きな領域に関しては、1 である分割変換フラグ値が暗示される。例えば、 $64 \times 64$  符号化単位のトップレベルを有する変換ツリーに関しては、サポートされた最大変換サイズが  $32 \times 32$  輝度サンプルである場合、4 つの  $32 \times 32$  領域への自動細分割 (すなわち: 符号化ビットストリーム 312 内で信号伝達されない) が発生する。

【0054】

右下の  $16 \times 16$  領域 642 は、輝度チャンネルに対する変換がない変換単位 (TU) (

50

10と番号付けされ、網掛けされている)を含んでおり、そのため、対応するゼロである符号化ブロックフラグ値716を有する。

【0055】

図6Cおよび図8は、4:2:2色差フォーマットに構成され、輝度チャンネルに対する変換ツリー630に対応し且つデータ構造800によって表現された色差チャンネルに対する1セットの変換を含む、色差チャンネルに対する、図6Aの例示的な変換ツリーに対応する例示的な変換ツリー630を図示する。変換ツリー階層が輝度チャンネルと色差チャンネルとの間の図6Aの構造によって共通するので、分割変換フラグ値は、データ構造700と800との間で共有される。データ構造700とは対照的に、データ構造800は、1である各変換分割フラグ値を有する符号化ブロックフラグ値を含む(すなわち変換ツリーの非リーフノード上に)。例えば、1である符号化ブロックフラグ値802は、変換分割フラグ702に関連づけられる。変換ツリーの非リーフノード上の符号化ブロックフラグ値がゼロである場合、子ノード上の符号化ブロックフラグ値は、ゼロとして暗示される(および、対応する符号化ブロックフラグは、符号化ビットストリーム312内で符号化されない)。非リーフ領域における符号化ブロックフラグ値は、たとえ大幅な残差係数が輝度チャンネル内にあり得るとしても、大幅な残差係数が子領域のどれにもなければ、各色差チャンネルに対する変換ツリーの下位レベルでの符号化ブロックフラグの符号化を終了することを可能にする。大部分の情報が輝度チャンネル内にあるので、これは典型的な取得されたフレームデータに共通の状態である。

【0056】

ビデオエンコーダ114およびビデオデコーダ134が4:4:4色差フォーマット用に構成される場合、変換単位(TU)サイズの所定のセットの1つでないサイズの任意の変換単位(TU)の各色差チャンネルの色差領域は、所定の変換単位(TU)の輝度領域と同一の寸法を有する(すなわち、暗黙的分割が行なわれない場合)。ビデオエンコーダ114およびビデオデコーダ134が4:4:4色差フォーマットに構成される場合、変換単位(TU)サイズの所定のセットの1つであるサイズの任意の変換単位(TU)の各色差チャンネルの色差領域は、所定の変換単位(TU)の輝度領域よりも小さな寸法を有する(すなわち、暗黙的分割が行なれる場合)。

【0057】

4:2:2色差フォーマットが使用されている場合、これは、各色差チャンネルに対して図6Cの色差サンプルの16x32領域662を含み、したがって、色差サンプルグリッド上の16x32領域を占有する、符号化単位(CU)602をもたらす。図6Cは、(図5Bとは対照的に)水平および垂直に等間隔に配置された各色差サンプルとともに、色差サンプルの配列として描画された、色差サンプルグリッド上の領域を図示する。4:2:2色差フォーマットの使用に起因して、図6Cの各色差領域は、図6Bの対応する輝度領域に対して水平に圧縮されたように見える。1である分割変換フラグ値702は、符号化単位(CU)602に対応する、16x32領域662を、8x16領域664などの4つの8x16領域に分割する。8x16領域664は、非正方形形状を有し、また、4x8領域670などの図6Cに図示された他の非正方形領域のサイズより大きい。各8x16領域に関しては、分割変換フラグ値704などの分割変換フラグ値は、輝度サンプル配列に対する変換ツリー630内にある四分木分割に類似した方式で、対応する8x16領域が4つのより小さな4x8領域にさらに細分割されるべきか否かを示す。右上の8x16領域672は、4つの4x8領域にさらに細分割される。1である符号化ブロックフラグ値804は、4つの4x8領域の各々が大幅な残差係数を含むかもしれないことを示す。各4x8領域に対する符号化ブロックフラグは、したがって、対応する領域に対する変換の有無を示すように要求される。これらの4つの4x8領域のうち、(網掛けされた)左下の4x8領域674は、変換単位(TU)を含むが、変換を含んでおらず、そのため、ゼロである符号化ブロックフラグ値814を有する。残りの4x8領域(領域670)は、各々、変換を有し、そのため、対応する1である符号化ブロックフラグ値を有する。左上の8x16領域は、2つの等しいサイズの8x8領域に細分割される。四分木細分

割とは対照的に、対応する分割変換フラグは、符号化ビットストリーム 3 1 2 内にない。

#### 【 0 0 5 8 】

符号化ビットストリーム 3 1 2 内にある信号伝達を行わずに、複数の領域（その各々は変換を有してもよい）への、変換単位（TU）の色差チャンネルなどのチャンネルの領域の分割は、「暗黙的分割」と呼ばれる。暗黙的分割は、このケース（ $8 \times 16$ ）のための非正方形変換をサポートするハードウェアを導入する必要性を省く。その代わりに、第 1 の  $8 \times 8$  変換 6 6 6 などの変換が用いられる。暗黙的分割から起因する領域の各々にとって、ゼロ残差情報をすべて含むことが可能であるので、暗黙的分割から起因する各領域内の変換の有無を指定することが必要である。従って、個別の符号化ブロックフラグ値は、暗黙的分割から起因する各領域に必要である。この場合、符号化ブロックフラグ値 8 0 6 および 8 0 8 は、第 1 の  $8 \times 8$  変換 6 6 6 および第 2 の  $8 \times 8$  変換 6 6 8 にそれぞれ対応する。暗黙的分割が行なわれない変換単位（TU）に関しては、各色差チャンネルに対する符号化ブロックフラグ値は、色差チャンネルに対する変換単位（TU）によって占有された領域に対する変換の有無を指定する。暗黙的分割を行なう場合、個別の符号化ブロックフラグ値（図 8 には図示せず）は、結果として生じる領域の各々に対して必要であるが、しかしながら、実施態様は、全体の変換単位（TU）に帰着可能な符号化ブロックフラグ値を保持してもよい。すべてのケースのうちの「1 つ」として個別の符号化ブロックフラグ値を暗示することができるかもしれないし、または、分割から起因する各領域の符号化ブロックフラグ値に対する論理的「OR」動作を実行することによって、個別の符号化ブロックフラグ値を決定することができるかもしれない。個別の符号化ブロックフラグ値が分割に起因する各領域の符号化ブロックフラグ値から決定される場合、個別の符号化ブロックフラグ値は、エントロピーエンコーダ 3 2 4 によって符号化ビットストリーム 3 1 2 内で符号化され、付加的な符号化ブロックフラグ（図 9 には図示せず）としてエントロピーデコーダ 4 2 0 によって符号化ビットストリーム 3 1 2 から復号化されてもよい。このようなケースにおいて、個別の符号化ブロックフラグ値がゼロである場合、分割から各領域の符号化ブロックフラグ値はゼロであると推論されてもよく、個別の符号化ブロックフラグ値が 1 である場合、分割から各々の領域に対する符号化ブロックフラグは、エントロピーエンコーダ 3 2 4 によって符号化ビットストリーム 3 1 2 内で符号化され、エントロピーデコーダ 4 2 0 によって符号化ビットストリーム 3 1 2 から復号化される。

#### 【 0 0 5 9 】

$16 \times 32$  領域 6 6 2 の左下の  $8 \times 16$  領域 6 8 0 は、 $8 \times 8$  変換が上部の  $8 \times 8$  暗示領域 6 8 2 内にあるが、 $8 \times 8$  変換が下の  $8 \times 8$  暗示領域 6 8 4 内にない暗黙的分割を図示する。右下の（網掛けされた） $8 \times 16$  配列 6 7 6 は、変換単位（TU）を含むが、暗黙的分割から起因する一方の正方形の  $8 \times 8$  領域内の変換を含まず、そのため、ゼロである符号化ブロックフラグ値 8 1 0、8 1 2 を有する。

#### 【 0 0 6 0 】

2 つの色差チャンネルの有無は、各色差チャンネルに対する変換の有無を指定するために用いられる個別の符号化ブロックフラグ値により、図 6 C に表現された構造の複製をもたらす。この実施態様において、分割は、サイズ  $4 \times 8$  以外の色差に対する領域サイズに対して推測され、結果として、（領域 6 7 0 内に含まれる） $4 \times 8$  変換 8 1 6 などの  $4 \times 8$  の長方形変換を用いて他のケース（例えば  $8 \times 8$ 、 $16 \times 16$ ）における既存の正方形変換の再使用を可能にする。従って、 $8 \times 16$  および  $16 \times 32$  などの 1 セットの所定の領域サイズは、2 つの領域への分割のために存在すると言ってもよく、したがって、（サイズ  $8 \times 8$  および  $16 \times 16$  の）2 つの変換を用いることができる。また、推測分割が発生する領域サイズの所定のセットの個別の定義が可能であり、既存の正方形変換および長方形変換の個別の組み合わせが用いられることを可能にすることになる。また、分割を常に推測することはある実施態様にとって可能であり、その場合には、長方形変換は、色差 4 : 2 : 2 色チャンネルに対して導入されない。このようなケースにおいて、推測分割が発生する領域サイズの所定のセットは、あらゆる色差領域サイズ（例えば、4 : 2 : 2 色差フォ

10

20

30

40

50

ーマットに対する  $4 \times 8$ 、 $8 \times 16$ 、および  $16 \times 32$ 、または  $4 : 4 : 4$  色差フォーマットに対する  $4 \times 4$ 、 $8 \times 8$ 、 $16 \times 16$ 、および  $32 \times 32$  ) を含んでいる。

$4 : 2 : 0$  色差フォーマットが使用中である場合、推測分割は、変換単位 (TU) 内のいずれの色差領域に対しても行なわれないので、各色差チャンネルに対する最大変換数は、常に 1 である (各色差チャンネルに対する符号化ブロックフラグ値は、色差変換が発生するかどうかをコントロールする)。

#### 【0061】

ビデオエンコーダ 114 およびビデオデコーダ 134 は、輝度と色差チャンネルとの間の差分と無関係に記載され、色差フォーマットから起因する、異なるサンプルグリッドは、モジュール内の差分の必要性を要する。実際的な実施態様は、輝度チャンネルおよび色差チャンネルに対する個別の「処理経路」を有してもよい。このような実施態様は、したがって、輝度サンプルおよび色差サンプルの処理を分離してもよい。符号化ビットストリーム 312 が輝度チャンネルおよび色差チャンネルの両方に対して単一のビットストリームであるので、エントロピーエンコーダ 324 およびエントロピーデコーダ 420 は分離されない。その上、フレームバッファ 332、432 などの単一のフレームバッファは、輝度サンプルおよび色差サンプルを保持しており、したがって分離されない。但し、モジュール 322 ~ 330 および 334 ~ 340、モジュール 422 ~ 430 および 434 は、実施態様が分離された輝度処理および色差処理を有し、輝度および色差に対して個別のロジックを有することを可能にして、それにより、「輝度処理経路」および「色差処理経路」を生成してもよい。

#### 【0062】

ある実施態様は、2つの  $16 \times 16$  領域への変換単位 (TU) の色差チャンネルの  $16 \times 32$  領域に対する分割を推測してもよいが、 $8 \times 16$  および  $4 \times 8$  のケースに対する分割を推測しなくてもよい。このような実施態様は、当該技術分野において確立された  $4$ 、 $8$  または  $16$  ポイントの変換ロジックに頼ることができる代わりに、色差処理経路に対して  $32$  ポイントの変換ロジックを導入する必要性を回避する。

#### 【0063】

図 9A および図 9B は、変換ツリーの階層レベルを符号化するか、さもなければ表現するために用いることができる構文構造を図示する。変換ツリーの非リーフノードにおいて、変換ツリーに対応する符号化ビットストリーム 312 の一部にある構文要素を定義するために、構文構造 900 は、データ構造 700、800 などのデータ構造によって再帰的に拡大される。変換ツリーのリーフノード (変換ツリー内で細分割が行なわれない) において、構文構造 930 は、符号化ビットストリーム 312 の一部にある構文要素を定義する。通常は、付加データ構造は、アルファチャンネルまたは深度マップを符号化するためなどに可能であるが、輝度のための 1 つのデータ構造および色差のための 2 つのデータ構造がある。または、少数のデータ構造は、単一のデータ構造が色差チャンネルによって共有され、符号化ブロックフラグ値が色差チャンネルの間で共有されることができるケースなどで利用されてもよい。変換ツリーの非リーフノード構文構造 902 は、変換ツリー 630 などの変換ツリーの 1 つの階層レベルの符号化を定義する。分割変換フラグ 910 は、分割変換フラグ値 702 などの、1 である分割変換フラグ値を符号化する。この値は、変換ツリー非リーフノード構文構造 902 が変換ツリー非リーフノード構文構造 902 または変換ツリーリーフノード構文構造 932 の付加的な実例を含む低階層レベル、または「子ノード」を含むことを示す。符号化ブロックフラグ 912 は、「U」色差チャンネルに対して 1 つの符号化ブロックフラグ値 802 を符号化し、符号化ブロックフラグ 914 は、「V」色差チャンネルに対するさらなる符号化ブロックフラグ値を符号化する。変換ツリー非リーフノード構文構造 902 が変換ツリー階層のトップレベルを定義しているのであれば、符号化ブロックフラグ 912、914 が存在する。変換ツリー非リーフノード構文構造 902 が変換ツリー階層のトップレベルを定義していないのであれば、符号化ブロックフラグ 912、914 は、変換ツリー階層の親レベル内に対応する符号化ブロックフラグが存在して一価であれば、単に存在するだけである。(最上部の階層レベルに比べて) より低

い階層レベルが変換ツリー 630 内に存在するとき、四分木細分割は行なわれる。この細分割は、変換ツリー非リーフノード構文構造 902 内に含まれる 4 つの変換ツリー構文構造 916、918、920、922 をもたらす。

【0064】

構文構造 930 は、変換ツリーリーフノード 932 のリーフノードの符号化を定義する（すなわち、それ以上細分割が行なわない場合）。分割変換フラグ 940 は、分割変換フラグ値 704 などの 0 である分割変換フラグ値を符号化する。

【0065】

対応する領域が最小サイズより大きい場合、分割変換フラグは、単に符号化される。例えば、領域 636 は、（サポートされた最小輝度変換サイズに対応する） $4 \times 4$  輝度サンプルの領域に対する容認できる最小サイズを有する。従って、変換分割フラグ値 714 は、ゼロとして推測され、分割変換フラグは、対応する変換ツリー構文構造に対して符号化されない。

10

【0066】

領域 636 に関しては、色差残差サンプルは、 $4 \times 8$  色差変換を用いて変換され、従って、推測された変換分割は存在しない。符号化ブロックフラグ 942 および符号化ブロックフラグ 946 などの符号化ブロックフラグは、色差チャンネルの各々に対する変換の有無を信号伝達するために存在してもよい。符号化ブロックフラグ 950 は、輝度チャンネルに対する変換の有無を信号伝達する。輝度チャンネルおよび色差チャンネルのための残差係数（もし存在すれば）は、変換単位（TU）構文構造 952 内に存在する。符号化ブロックフラグ 950 の値が 1 であるならば、輝度残差ブロック 954 は、符号化ビットストリーム 312 内にある。各色差チャンネルに対する符号化ブロックフラグの値が 1 であるならば、対応する色差残差ブロック 956、960 は、符号化ビットストリーム 312 内に存在する。

20

【0067】

領域 664 に関しては、色差残差サンプルが 2 つの  $8 \times 8$  色差変換を用いて変換され、従って、推測された変換分割は存在する。符号化ブロックフラグ 942、946 は、もし存在すれば、第 1 の  $8 \times 8$  変換 666 の各色差チャンネルに対して  $8 \times 8$  変換の有無を信号伝達する。符号化ブロックフラグ 944、符号化ブロックフラグ 948 は、もし存在すれば、第 2 の  $8 \times 8$  変換 668 の各色差チャンネルに対して  $8 \times 8$  変換の有無を信号伝達する。符号化ブロックフラグ 944 の値が 1 であるならば、色差残差ブロック 958 は、符号化ビットストリーム 312 内に存在する。符号化ブロックフラグ 948 の値が 1 であるならば、色差残差ブロック 962 は、符号化ビットストリーム 312 内に存在する。

30

図 9B に図示されるような構文構造 930 は、推測された変換分割に隣接して符号化された各色差チャンネルの第 1 および第 2 の変換を示す。隣接する各色差チャンネルに対して構文要素を符号化するか、または他の構文要素とともに散在した各色差チャンネルに対して構文要素を符号化する他の配列が、または用いられてもよい。

【0068】

図 9C および図 9D は、変換ツリーの階層レベルを符号化するか、さもなければ表現するために用いることができる、代替的な構文構造 9100 を図示する。変換ツリーの非リーフノードにおいて、変換ツリーに対応する符号化ビットストリーム 312 の一部内にある構文要素を定義するために、代替的な構文構造 9100 は、データ構造 700、800 などのデータ構造に従って再帰的に拡大される。代替的な構文構造 9100 のインスタンスは、各変換単位（TU）を含む、リーフノードを含む変換ツリー内の各々のノードごとに存在する。各色差チャンネルに対する変換単位（TU）を細分化するために「推測分割」が発生する場合、構文構造 9130 は、符号化ビットストリーム 312 の一部内にある構文要素を定義する。通常は、付加データ構造は、アルファチャンネルまたは深度マップを符号化するためなどに可能であるが、輝度のための 1 つのデータ構造および色差のための 2 つのデータ構造がある。または、少数のデータ構造は、単一のデータ構造が色差チャンネルによって共有され、符号化ブロックフラグ値が色差チャンネルの間で共有されることができ

40

50

るケースなどで利用されてもよい。変換ツリー構文構造 9 1 0 2 は、変換ツリー 6 3 0 などの変換ツリーの 1 つの階層レベルの符号化を定義する。

【 0 0 6 9 】

変換ツリー 6 3 0 などの変換ツリーの非リーフノードにおける変換ツリー構文構造 9 1 0 2 のインスタンスに関しては、分割変換フラグ 9 1 1 0 は、分割変換フラグ値 7 0 2 などの 1 である分割変換フラグ値を符号化する。この値は、変換ツリー構文構造 9 1 0 2 のインスタンスが変換ツリー構文構造 9 1 0 2 または「子ノード」の付加的な実例を含む、低い階層レベルを含むことを示す。符号化ブロックフラグ 9 1 1 2 は、符号化ブロックフラグ 9 1 2 の記述によって符号化ブロックフラグ値を符号化する。符号化ブロックフラグ 9 1 1 4 は、符号化ブロックフラグ 9 1 4 の記述によって符号化ブロックフラグ値を符号化する。（最上部の階層レベルに比べて）より低い階層レベルが変換ツリー 6 3 0 内に存在するとき、四分木細分割は行なわれる。この細分割は、変換ツリーノード構文構造 9 1 0 2 内に含まれている、4 つの変換ツリー構文構造 9 1 1 6、9 1 1 8、9 1 2 0、9 1 2 2 をもたらす。変換ツリー構文構造 9 1 1 6、9 1 1 8、9 1 2 0、9 1 2 2 の各々は、変換ツリー構文構造 9 1 0 2 の別のインスタンスである。符号化ブロックフラグ 9 1 2 4 および輝度変換単位部分 9 1 2 6 は、変換ツリー構文構造 9 1 0 2 から存在しないことになる。

10

【 0 0 7 0 】

実施態様は、また、変換ツリー構文構造 9 1 0 2 を符号化ブロックフラグ 9 1 2 4 および輝度変換単位部分 9 1 2 6（もし存在すれば）が、符号化ブロックフラグ 9 1 1 4 と変換ツリー構文構造 9 1 1 6 との間などの、変換ツリー構文構造 9 1 0 2 内に初期に配置されるように配置してもよい。

20

【 0 0 7 1 】

変換ツリー 6 3 0 などの変換ツリーのリーフノードにおける変換ツリー構文構造 9 1 0 2 のインスタンスに関しては、分割変換フラグ 9 1 1 0 は、分割変換フラグ値 7 0 4 などのゼロである分割変換フラグ値を符号化する。変換ツリー構文構造 9 1 0 2 のインスタンスは、したがって、変換ツリー 9 3 0 内の変換単位 (TU) に対応する。変換単位 (TU) は、符号化単位 (CU) 6 0 2 などの変換単位 (CU) および変換深度を含む符号化単位 (CU) によって決定されたサイズを有する。符号化ブロックフラグ 9 1 1 2 は、「U」色差チャンネルに対する推測分割に起因するあらゆる色差領域が 1 である符号化ブロックフラグ値を有してもよいことを示すために、1 である符号化ブロックフラグ値を符号化する。符号化ブロックフラグ 9 1 1 2 がゼロの値を符号化するならば、「U」色差チャンネルに対する推測分割に起因する各色差領域に対する符号化ブロックフラグ値は、ゼロとして推測された符号化ブロックフラグ値を有する。符号化ブロックフラグ 9 1 1 2 が 1 である値を符号化する場合でさえ、実施態様は、推測分割から起因する各色差領域に対してゼロである値を有する符号化ブロックフラグをさらに符号化してもよい。そのため、実施態様は、符号化ビットストリーム 3 1 2 から符号化ブロックフラグ 9 1 1 2 を省略してもよいが、その代り、省略された符号化ブロックフラグ 9 1 1 2 に対して 1 つの符号化ブロックフラグ値を常に推測する。符号化ブロックフラグ 9 1 1 4 は、符号化ブロックフラグ 9 1 1 2 と同様の方式で「V」色差チャンネルに対してさらなる符号化ブロックフラグ値を符号化する。4 つの色差領域への推測分割が発生するものに一致する変換単位 (TU) サイズに関して（色差残差係数配列の最大数は 4 である）、4 つの変換ツリー構文構造 9 1 1 6、9 1 1 8、9 1 2 0、9 1 2 2 は、変換ツリーノード構文構造 9 1 0 2 内に含まれる。2 つの色差領域への推測分割が発生するものに一致する変換単位 (TU) サイズに関して（色差残差係数配列の最大数は 2 である）、変換ツリー構文構造 9 1 1 6、9 1 1 8 などの 2 つの変換ツリー構文構造は、変換ツリーノード構文構造 9 1 0 2 内に含まれる。変換ツリー構文構造 9 1 1 6、9 1 1 8、9 1 2 0、9 1 2 2 の各々は、色差構文構造 9 1 3 2 に対する変換ツリーのインスタンスである。符号化ブロックフラグ 9 1 2 4 は、変換単位 (TU) の輝度チャンネルに対する変換の欠如の有無を指定する、符号化ブロックフラグ値 7 0 8 などの符号化ブロックフラグ値を符号化する。変換単位 9 1 2 6 の輝度部分は、輝

30

40

50

度残差構文要素 9 1 2 8 として輝度残差係数配列を符号化する。

【 0 0 7 2 】

推測分割が行なわれる場合に各色差領域対して存在するのみの、色差構文構造 9 1 3 2 に対する変換ツリーは、変換ツリー構文構造 9 3 0 の構文の縮小されたセットを含む。符号化ブロックフラグ 9 1 4 2 は、色差領域の「U」色差チャンネルに対する符号化ブロックフラグ値を符号化する。符号化ブロックフラグ 9 1 4 4 は、色差領域の「V」色差チャンネルに対する符号化ブロックフラグ値を符号化する。変換単位 (TU) 9 1 4 6 の色差部分は、変換単位 (TU) 構文構造 9 5 2 のサブセットを符号化する。符号化ブロックフラグ 9 1 4 2 の値が 1 であるならば、変換単位 (TU) 9 1 4 6 の色差部分は、「U」色差チャンネルに対する色差残差構文要素 9 1 5 0 として色差残差係数配列を符号化する。符号化  
10  
ブロックフラグ 9 1 4 4 の値が 1 であるならば、変換単位 (TU) 9 1 4 6 の色差部分は、「V」色差チャンネルに対する色差残差構文要素 9 1 5 2 として色差残差係数配列を符号化する。

【 0 0 7 3 】

図 9 D に図示されるような構文構造 9 1 3 0 は、隣接して符号化された第 1 および第 2 の符号化ブロックフラグに続く、推測された変換分割に対する各色差チャンネルの第 1 および第 2 の色差残差係数配列を示す。各色差チャンネルに隣接して符号化ブロックフラグおよび色差残差係数配列を符号化するなどの他の配列が、代わりに用いられてもよい。推測された変換分割は、2 つの  $8 \times 8$  領域への  $8 \times 16$  領域  $664$  分割により図示されるが、代替的な実施態様は、他の領域に対する分割を実行してもよい。例えば、いくつかの  
20  
実施態様は、2 つの  $16 \times 16$  領域への  $16 \times 32$  領域の分割を推測してもよい。このような実施態様は、色差処理経路内の 32 ポイントの 1 次元変換に対する必要性を有利に回避する。4 : 2 : 0 色差フォーマットが適用される場合、32 ポイントの 1 次元変換が色差処理経路には必要でないので、32 ポイントの 1 次元変換に対する要件は、色差処理経路から全体に取り除かれる。輝度および色差チャンネルを分離する個別の処理回路を用いる実施態様は、したがって、色差処理回路の低実装コストを達成することができる。

【 0 0 7 4 】

各輝度サンプル位置に対する 1 つの色差サンプル位置があるところに、4 : 4 : 4 色差フォーマットは存在する。従って、この形式により、色差チャンネルおよび輝度チャンネルに対する変換は、同一のサイズを有してもよい。輝度処理経路内の  $32 \times 32$  の最大変換サイ  
30  
ズにより、これは、分離された実装に対して色差処理経路内の  $32 \times 32$  変換を導入することを要するだろう。特定の実施態様は、色差処理経路内の現存の  $16 \times 16$  変換の再使用を可能にする、4 つの  $16 \times 16$  領域に、 $32 \times 32$  領域を分割するために各色差チャンネルに対する分割を推測してもよい。 $32 \times 32$  変換が 4 : 4 : 4 色差フォーマットに対して色差処理経路においてのみ用いられるので、4 つの  $16 \times 16$  領域への  $32 \times 32$  領域を分割する各々の色差チャンネルに対する分割の推測は、必要とされる処理回路を低減して、 $32 \times 32$  変換が色差処理経路から取り除かれるのを可能にする。このような実施態様は、各色差チャンネルに対する 4 つの符号化ブロックフラグ値、したがって、符号化ビットストリーム 3 1 2 内の各色差チャンネルに対して構文構造 9 3 0 内で符号化された 4 つ  
40  
の符号化ブロックフラグまで、を必要とするだろう。

【 0 0 7 5 】

4 : 2 : 2 色差フォーマットをサポートする実施態様は、また、各色差チャンネルが 4 つの  $8 \times 16$  領域に  $32 \times 16$  領域を分岐するために分割を推測してもよい。このような実施態様は、各色差チャンネルに対する 4 つの符号化ブロックフラグ値を要し、したがって、符号化ビットストリーム 3 1 2 内の各色差チャンネルに対して構文構造 9 3 0 内で符号化された 4 つの符号化ブロックフラグ、したがって「CU3」、「CU4」、「CV3」および「CV4」符号化ブロックフラグ (図 9 B には図示せず) は、変換単位 (TU) 構文構造 9 5 2 内に導入されてもよい。このような実施態様は、色差処理経路への 32 ポイントの変換ロジックの導入を回避するとともに、 $8 \times 16$  領域が細分割されない場合には、色差チャンネルに対してサイズ  $8 \times 16$  の変換を要する (輝度チャンネル内の) サイズ  $16 \times 1$   
50

6の変換単位(TU)に必要な $8 \times 16$ 変換ロジックを再使用してもよい。

【0076】

図10は、変換ツリー非リーフノード構文構造902および変換ツリーリーフノード構文構造932の符号化によって、変換単位(TU)を符号化する方法1000を示す概略フロー図である。方法1000は、変換単位(TU)の色差チャンネルを参照しながら説明するが、方法1000は、変換単位(TU)のあらゆる色差チャンネルに適用されてもよい。変換ツリー非リーフノード構文構造902および変換ツリーリーフノード構文構造932が、変換ツリー内の1つのノードを記載するため、方法1000は、符号化ビットストリーム312への変換ツリーの1つのノードを符号化する。例えば、方法1000は、ハードウェアにおいて、またはプロセッサ205上で実行可能なソフトウェアによって、実施されてもよい。方法1000は、変換ツリーのトップレベルに対して最初に起動され、変換ツリーの子ノードを符号化するためにそれ自体(再帰的に)起動することができる。変換単位サイズを決定するステップ1002は、変換ツリーおよび変換単位(TU)の変換深度値を含む符号化単位(CU)サイズに従って変換ツリーにおける変換単位(TU)のサイズを決定する。方法1000が変換ツリーのトップレベルで起動される場合、変換深度値は、ゼロに設定される。さもなければ、変換深度値は、方法1000の親インスタンスによって提供される。分割変換フラグ値702などの分割変換フラグ値は、変換深度値が変換深度を可能にする最大値未満であれば、分割変換フラグ910として符号化ビットストリーム312内で符号化される。

【0077】

分割変換フラグ値が1である場合、変換ツリー階層の親ノードが、1である対応する符号化ブロックフラグ値を有する場合にのみ、色差符号化ブロックフラグ912、914は、各色差チャンネルに対して符号化される。方法1000は、その後、変換ツリーの(変換ツリー構文構造916、918、920、922によって符号化ビットストリーム312の部分において表現された)各子ノードに対して方法1000の新規のインスタンスを起動する。子ノードに対して起動された方法1000の各インスタンスには、1インクリメントした現在の方法1000インスタンス変換深度値に等しい変換深度値が提供される。

【0078】

分割変換フラグ値がゼロである場合、順方向変換の最大数を識別するステップ1004は、符号化されている領域の各色差チャンネルに対して変換最大数(n)を決定する。推測分割が行なわれない場合、この数nは1になる。4:2:2色差フォーマットが使用され、 $8 \times 16$ 領域664などの色差チャンネルの長方形領域に接触し、領域サイズが所定の領域サイズ(例えば $16 \times 32$ 、 $8 \times 16$ など)のセットの1つである場合、推測分割は行なわれ、最大変換数は2になるだろう(さもなければ、変換数は1になる)。さもなければ(領域サイズが所定の領域サイズのセットの1つではない場合)最大変換数は1になる。例えば、 $4 \times 8$ が所定の領域サイズのセットの1つでなければ、最大変換数は1になる。4:4:4色差フォーマットが使用され、接触した領域サイズが所定の領域サイズ(例えば $32 \times 32$ 領域など)のセットの1つである場合、推測分割は行なわれ、最大変換数は4になる。さもなければ(領域サイズが所定の領域サイズのセットの1つではない場合)最大数は1になる。例えば、 $8 \times 8$ が所定の領域サイズのセットの1つでなければ、最大変換数は1になる。所定の領域サイズのセットは $8 \times 16$ を含むが、他の所定の領域サイズのセットは、4:2:2色差フォーマットが使用中のある場合に $16 \times 32$ 、または4:4:4色差フォーマットが使用中のある場合に $32 \times 32$ などの場合のみに可能である。

【0079】

各々の色差チャンネルに対して、親ノードが1つの符号化ブロックフラグ値を有する場合、nの各々に対して、符号化ブロックフラグは、符号化ビットストリーム312内で符号化される。例えば、変換数が2に等しい場合、符号化ブロックフラグ942、944は、分割によって推測された2つの領域の各々に対する変換の有無を示す。順方向変換を選択



するステップ1006は、変換深度に順番に依存し、したがって、最大符号化単位内の変換単位の階層レベルに関係する、変換単位(TU)のサイズに基づいて、最大変換数の各々に対して、所定の順方向変換のセットから順方向変換を選択する。変換深度がゼロに等しい場合、変換単位(TU)サイズは、符号化単位(CU)サイズに等しい。変換深度の各インクリメントに対して、変換単位(TU)サイズは半分になる。32×32符号化単位(CU)のサイズ、ゼロである変換深度、および4:2:2色差フォーマットを用いることに関しては、変換単位(TU)サイズは、したがって32×32になり、色差に対する変換サイズは、したがって16×32になる。例えば、最大変換数が2で、色差に対する領域サイズが16×32である場合、16×16順方向変換は、推測分割に起因する色差に対して16×16領域の各々に対して選択される。

10

#### 【0080】

順方向変換を適用するステップ1008は、1である符号化ブロックフラグ値を有する、対応する領域上で最大変換数の各々に対して順方向変換を実行する。色差残差サンプル配列を符号化するステップ1008は、一般的に、変換モジュール320によって実行される。これは、色差残差係数配列(周波数領域表現)への各色差残差サンプル配列(空間領域表現)の変換をもたらす。

#### 【0081】

色差残差係数配列を符号化するステップ1010は、1である符号化ブロックフラグ値を有する各色差チャンネルの変換領域の最大数の各々に対して、色差残差係数配列を符号化ビットストリーム312に符号化する。所定の色差チャンネルについて所定の変換単位に対して符号化された色差残差係数配列の数は、各変換の符号化ブロックフラグ値に依存し、したがって、ゼロから(多くても)最大変換数まで変化することになる。例えば、変換数が2であり、双方の色差チャンネルがカウント値の各々に対して1である符号化ブロックフラグ値を有する場合、色差残差ブロック956、958、960、962は、符号化ビットストリーム312内で符号化される。所定の色差チャンネルに関して各変換に対する符号化ブロックフラグ値がゼロであるならば、色差残差ブロックは、その色差チャンネルに対する符号化ビットストリーム312内で符号化されない。色差残差係数配列を符号化するステップ1010は、一般的に、エン트로ピーエンコーダ324によって実行される。

20

図11は、変換ツリー非リーフノード構文構造902および変換ツリーリーフノード構文構造932の復号化によって、変換単位(TU)を復号化する方法1100を示す概略フロー図である。方法1100は、変換単位(TU)の色差チャンネルを参照して説明するが、方法1100は、変換単位(TU)のあらゆる色差チャンネルに適用されてもよい。変換ツリー非リーフノード構文構造902および変換ツリーリーフノード構文構造932が、変換ツリー内の1つのノードを記載するため、方法1100は、符号化ビットストリーム312から変換ツリーの1つのノードを復号化する。方法1100は、適切なハードウェア、または代わりに例えばプロセッサ205によって実行可能なソフトウェアにおいて、実行されてもよい。方法1100は、変換ツリーのトップレベルに対して最初に起動され、変換ツリーの子ノードを復号化するためにそれ自体(再帰的に)起動することができる。変換単位(TU)サイズを決定するステップ1102は、変換単位サイズを決定するステップ1002と同一の方式で変換単位(TU)サイズを決定する。変換単位サイズを決定するステップ1102は、変換ツリーおよび変換単位(TU)の変換深度値を含む符号化単位(CU)サイズに従って変換ツリーにおける変換単位(TU)のサイズを決定する。方法1100が変換ツリーのトップレベルで起動される場合、変換深度値は、ゼロに設定されるか、さもなければ、変換深度値は、方法1100の親インスタンスによって提供される。分割変換フラグ値702などの分割変換フラグ値は、変換深度値が変換深度を可能にする最大値未満であれば、分割変換フラグ910として符号化ビットストリーム312から復号化される。

30

40

#### 【0082】

分割変換フラグ値が1である場合、変換ツリー階層の親ノードが、1である対応する符号化ブロックフラグ値を有する場合にのみ、色差符号化ブロックフラグ912、914は

50

、各色差チャンネルに対して復号化される。方法 1 1 0 0 は、その後、変換ツリーの（変換ツリー構文構造 9 1 6、9 1 8、9 2 0、9 2 2 によって符号化ビットストリーム 3 1 2 の部分において表現された）各子ノードに対して方法 1 1 0 0 の新規のインスタンスを起動する。子ノードに対して起動された方法 1 1 0 0 の各インスタンスには、1 インクリメントした現在の方法 1 1 0 0 インスタンス変換深度値に等しい変換深度値が提供される。

#### 【 0 0 8 3 】

分割変換フラグ値がゼロである場合、逆変換の最大数を識別するステップ 1 1 0 4 は、順方向変換の最大数（ $n$ ）を識別するステップ 1 0 0 4 と同一の方式で、復号化されている領域の各色差チャンネル内にある少なくとも 1 つの色差残差係数配列の各々に対して（最大）変換数（ $n$ ）を決定する。推測分割が行なわれない場合、この数  $n$  は 1 になる。4 : 2 : 2 色差フォーマットが使用されており、 $8 \times 16$  領域 6 6 4 などの色差チャンネルの長方形領域に接触し、領域サイズが所定の領域サイズ（例えば  $16 \times 32$ 、 $8 \times 16$  など）のセットの 1 つである場合、推測分割は行なわれ、最大変換数は 2 なる（さもなければ、変換数は 1 になる）。さもなければ（領域サイズが所定の領域サイズのセットの 1 つではない場合）最大変換数は 1 になる。例えば、 $4 \times 8$  が所定の領域サイズのセットの 1 つでなければ、最大変換数は 1 になる。4 : 4 : 4 色差フォーマットが使用されており、接触した領域サイズが所定の領域サイズ（例えば  $32 \times 32$  領域など）のセットの 1 つである場合、推測分割は行なわれ、最大変換数は 4 になる。さもなければ（領域サイズが所定の領域サイズのセットの 1 つではない場合）最大数は 1 になる。例えば、 $8 \times 8$  が所定の領域サイズのセットの 1 つでなければ、最大変換数は 1 になる。所定の領域サイズのセットは  $8 \times 16$  を含むが、他の所定の領域サイズのセットは、4 : 2 : 2 色差フォーマットが使用されている場合は  $16 \times 32$ 、または 4 : 4 : 4 色差フォーマットが使用されている場合は  $32 \times 32$  などの場合のみに可能である。各々の色差チャンネルに対して、親ノードが符号化ブロックフラグ値 1 を有する場合、（ $n$  個）の変換各々に対して、符号化ブロックフラグは、符号化ビットストリーム 3 1 2 内で復号化される。例えば、最大変換数が 2 に等しい場合、符号化ブロックフラグ 9 4 2、9 4 4 は、分割によって推測された 2 つの領域の各々に対する変換の有無を示す。

#### 【 0 0 8 4 】

色差残差係数配列を復号化するステップ 1 1 0 6 は、その後、符号化ブロックフラグ値 1 を有する符号化ビットストリーム 3 1 2 から各色差チャンネルの最大変換数領域の各々に対して残差係数配列を復号化する。所定の色差チャンネルについて所定の変換単位に対して復号化された色差残差係数配列の数は、各変換の符号化ブロックフラグ値に依存し、したがって、ゼロから（多くても）「変換数（ $n$ ）」まで変化することになる。例えば、変換数が 2 であり、双方の色差チャンネルがカウント値の各々に対して 1 である符号化ブロックフラグを有する場合、色差残差ブロック 9 5 6、9 5 8、9 6 0、9 6 2 は、符号化ビットストリーム 3 1 2 から復号化される。色差残差係数配列を復号化するステップ 1 1 0 6 は、一般的に、1 である符号化ブロックフラグ値を有する各色差残差係数配列についてエントロピーデコーダ 4 2 0 によって実行される。

#### 【 0 0 8 5 】

逆変換を選択するステップ 1 1 0 8 は、その後、各色差チャンネルに対して 1 である符号化ブロックフラグ値を有する最大変換数の各々に対して、所定の逆変換のセットから逆変換を選択する。例えば、最大変換数が 2 であり、領域サイズが  $16 \times 32$  であり、2 つの変換の各々に対する符号化ブロックフラグ値が 1 である場合、 $16 \times 16$  逆変換が、推測分割から起因する  $16 \times 16$  領域の各々に対して選択される。

#### 【 0 0 8 6 】

逆変換を適用するステップ 1 1 1 0 は、符号化ブロックフラグ値 1 を有する、対応する領域上の最大変換数の各々に対して逆変換を実行する。これは、復号化されたビデオフレームを表現する色差残差サンプル配列（空間領域表現）への各色差残差係数配列（周波数領域表現）の変換をもたらす。逆変換を適用するステップ 1 1 1 0 は、一般的に、逆スケ

10

20

30

40

50

ール / 変換モジュール 4 2 2 によって実行される。

【 0 0 8 7 】

図 1 2 A は、対角スキャンパターン 1 2 0 1 を示し、図 1 2 B は、水平スキャンパターン 1 2 0 2 を示し、図 1 2 C は、垂直走査パターン 1 2 0 3 を示しており、各々は  $4 \times 8$  変換単位 1 2 0 0 に対するスキャンパターンを示す。図示されたスキャンパターンを用いて  $4 \times 8$  変換単位 1 2 0 0 をスキャンするそれらの実施態様は、残差係数が「サブブロック」として知られている  $4 \times 4$  ブロックにおいてグループ化される特性を有する。そのため、符号化ビットストリーム 3 1 2 内にある「係数グループ」フラグは、各サブブロックに対して、少なくとも 1 つの有意な（ゼロでない）残差係数の有無を示すために用いられてもよい。 $4 \times 8$  変換に対する  $4 \times 4$  サブブロックサイズの適用は、係数がサブブロックに常にグループ化される、他の変換サイズ中のスキャンパターンとの整合性を実現する。

10

【 0 0 8 8 】

特定の実施態様は、各サブブロック内の少なくとも 1 つのゼロでない残差係数の有無を信号伝達するために係数グループフラグを適用してもよい。有利なことに、これらのスキャンパターンは、すべての変換サイズに対してサブブロック処理を再使用することによって、残差係数を処理する制御ソフトウェアまたはデジタル回路の再使用を可能にする。用いられる特定のスキャンパターンは、配置された予測単位 (PU) のイントラ予測方向などの基準に従って選択されてもよい。変換により  $4 : 2 : 2$  色差フォーマットサンプルグリッド上の色差サンプルを符号化する場合、各色差サンプルが、輝度サンプルの非正方形 ( $2 \times 1$ ) 配列にマッピングされるためイントラ予測モードの「方向」または角度に影響を与えるので、イントラ予測方向とスキャンパターンとの間の関連性が変更される。スキヤニングは、変換単位 (TU) の左上隅に配置された、DC 係数にて終了する「逆」方向に示される。さらに、スキヤニングは、変換単位 (TU) の右下コーナーから開始するようにする必要はない。変換単位 (TU) の左上領域内のゼロでない残差係数の優位性に起因して、スキヤニングは、「最後の有意係数位置」から開始して、左上係数に達する方向で逆方向に進んでもよい。

20

【 0 0 8 9 】

他の実施態様は、残差係数を符号化するために所定の領域に対して単一のスキャンを適用し、その後、これらの残差係数に対して 1 つ以上の変換を適用してもよい。この場合、1 つの符号化ブロックフラグのみが領域に対して、従ってスキャンパターンによってカバーされるすべての変換に対して、用いられる。少なくとも 1 つの有意な残差係数がいずれかのスキャン内に存在する場合、符号化ブロックフラグは 1 に設定される。例えば、図 1 2 A ~ 図 1 2 C の  $4 \times 8$  スキャンパターンは、2 つの  $4 \times 4$  変換の残差係数を符号化するために適用されてもよい。残差係数の 2 つの  $4 \times 4$  配列は、スキャンパターンに適している  $4 \times 8$  配列を形成するために連結されてもよい。単一のスキャンが配列を通じて実行されるので、単一の「最後の有意係数」の位置は、スキャンパターンに対してビットストリーム内で符号化され、単一の符号化ブロックフラグ値は、配列にとって十分である。修正離散コサイン変換 (DCT) のエネルギー圧縮性は、長方形の係数配列へのスキャンパターンの経路に沿って各正方形変換の係数を交互配置するなどの他のスキームに対する有利性を与える。これは、各  $4 \times 4$  残差係数配列内の残差係数値の密度が、エントロピーデコード 4 2 0 による後続の復号化に対して、高圧縮効率がエントロピーエンコード 3 2 4 によって生成されることを可能にする、組み合わせられた  $4 \times 8$  配列とほぼ等しくなるという有利性を与える。

30

40

【 0 0 9 0 】

色差の色チャンネルを符号化するある実施態様は、 $4 : 2 : 0$  色差サンプルグリッドに対応する色差サンプル位置で残差サンプルを符号化する第 1 の変換と、 $4 : 2 : 0$  色差サンプルグリッドに対して、 $4 : 2 : 2$  色差サンプルグリッド内に導入された付加的な色差サンプル位置で残差サンプルを符号化する第 2 の変換とを用いてもよい。このような実施態様は、第 2 の変換の出力が第 1 の変換にたいして、第 2 の変換に対する残差サンプルを生

50

成するために残差サンプルに加えられる（さもなければ組み合わせられる）、アダマール変換などの、第2の変換に対して単純化された変換を有利に用いてもよい。有利には、ハール変換などの変換を実施する前処理段階は、4:2:0色差フォーマットに対する色差サンプルグリッドに、4:2:2色差フォーマットに対する色差サンプルグリッドをサンプリングするために用いられてもよい。このような構成はサイド情報として前処理段階から付加的な残差係数を伝送しなければならない。このような残差は、前処理変換が最大符号化単位（LCU）レベルで適用されるケースにおいて各最大符号化単位（LCU）に適用される。

#### 【0091】

所定の領域に対する複数の変換を有する実施態様は、全体領域をカバーする単一の組み合わせられたスキャンまたは個別のスキャンのいずれかを各々の変換に対して用いてもよい。複数の変換に対するスキヤニングが単一のスキャンに組み合わせられた場合、1つの符号化ブロックフラグのみが、スキャンされる各領域に必要なものである。単一の組み合わせられたスキャンを用いるそれら実施態様は、同様の分光特性を有する各変換から残差係数を配置するために、係数ごとに基づいて交互配置するなど、各変換の残差係数を交互配置することによって、残差係数の高圧縮を実現してもよい。

#### 【0092】

付録Aは、構文構造900および構文構造930に関連する開発中の高効率動画像符号化方式（HEVC）規格にありうる「テキスト」を図示する。付録Aにおけるtransform\_tree（）関数の各インスタンスは、図9Aおよび図9Cにおいて「TT」とラベルづけされた構文構造の一部として表現され、付録Aにおけるtransform\_unit（）関数の各インスタンスは、図9Aおよび図9Bにおいて「TU」とラベルづけされた構文構造の一部として表現される。付録A内に提示されるテキストは、構文構造900、930に対応するテキストの一例であり、他の例も可能である。構文構造900、930に対応するテキストは、ビデオエンコーダ114がビットストリームを符号化する方法1000を実行し、ビデオデコーダ134がビットストリームを復号化する方法1100を実行する、ということを暗に示す。

#### 【0093】

付録Bは、構文構造9100および構文構造9130に関連する開発中の高効率動画像符号化方式（HEVC）規格にありうる「テキスト」を図示する。付録Bにおけるtransform\_tree（）関数の各インスタンスは、図9Cおよび図9Dにおいて「TT」とラベルづけされた構文構造の一部として表現され、付録Aにおけるtransform\_unit（）関数の各インスタンスは、図9Cおよび図9Dにおいて「TU」とラベルづけされた構文構造の一部として表現される。付録B内に提示されるテキストは、構文構造9100、9130に一致するテキストの一例であり、他の例も可能である。構文構造9100、9130に対応するテキストは、ビデオエンコーダ114がビットストリームを符号化する方法1000を実行し、ビデオデコーダ134がビットストリームを復号化する方法1100を実行する、ということを暗に示す。

#### 【0094】

付録Aおよび付録B内のテキストは、結果として実装され、それによって、4:4:4色差フォーマットに対して構成されたサイズ32×32の変換単位（TU）において接する32×32色差領域は、適用される（最大数の）4つの16×16色差変換をもたらす。付録Aおよび付録B内のテキストに起因する実装は、より小さなサイズの変換単位（TU）に適用されて4:2:2色差フォーマットに対して構成された場合、（最大の）1つの色差変換が適用される。例えば、8×16変換は、8×16色差領域に適用され、4×8変換は、4×8色差領域に対する適用される。

#### 【産業上の利用可能性】

#### 【0095】

10

20

30

40

50

記載の構成は、コンピュータおよび情報処理産業に適用可能であり、特にビデオ信号などの信号の符号化および復号化のためのデジタル信号処理に適用可能である。

前述のものは、本発明のいくつかの実施形態のみを記載しており、変形および/または変更は、本発明の範囲および精神、例示的であり限定的でない実施形態から逸脱せずに行なうことができる。

【 0 0 9 6 】

(オーストラリアのみ)この明細書との関連において、単語「備える (comprising)」は、「主として但し必ずではなく単に含む (including)」または「有する (having)」または「含む (including)」ことを意味し、「だけから構成される (consisting only of)」ことは意味しない。「備える (comprise)」および「備える (comprises)」などの単語「備える」の多様性は、対応する多様の意味を有する。

10

【 0 0 9 7 】

[ 付 録 A ]

transform\_\_tree ( ) および transform\_\_unit ( ) は、ループ構成体を用いて推測された色差分割を実施する

### 7 . 3 . 1 1 変換ツリー構文

transform_tree( x0, y0, xBase, yBase, log2TrafoSize, trafoDepth, blkIdx ) {	記述子
if( log2TrafoSize <= Log2MaxTrafoSize && log2TrafoSize > Log2MinTrafoSize && trafoDepth < MaxTrafoDepth && !(IntraSplitFlag && trafoDepth == 0) )	
split_transform_flag[ x0 ][ y0 ][ trafoDepth ]	ae(v)
if( trafoDepth == 0    log2TrafoSize > 2 ) {	
if( trafoDepth == 0    cbf_cb[ xBase ][ yBase ][ trafoDepth - 1 ] ) {	
for( tIdx = 0; tIdx < TrafoCrCbCnt; tIdx++ ) {	
cbf_cb[ x0 + ( ( 1 << log2CrCbTrafoHorSize ) * ( tIdx mod TrafoCrCbHorCnt ) ][ y0 + ( 1 << log2CrCbTrafoVertSize ) * ( tIdx div TrafoCrCbVertCnt ) ][ trafoDepth + ( TrafoCrCbCnt > 1 ) ]	ae(v)
}	
cbf_cb[ x0 ][ y0 ][ trafoDepth ]  = (TrafoCrCbCnt > 1)	
}	
if( trafoDepth == 0    cbf_cr[ xBase ][ yBase ][ trafoDepth - 1 ] ) {	
for( tIdx = 0; tIdx < TrafoCrCbCnt; tIdx++ ) {	
cbf_cr[ x0 + ( ( 1 << log2CrCbTrafoHorSize ) * ( tIdx mod TrafoCrCbHorCnt ) ][ y0 + ( 1 << log2CrCbTrafoVertSize ) * ( tIdx div TrafoCrCbVertCnt ) ][ trafoDepth + ( TrafoCrCbCnt > 1 ) ]	ae(v)
}	
cbf_cr[ x0 ][ y0 ][ trafoDepth ]  = (TrafoCrCbCnt > 1)	
}	
}	
if( split_transform_flag[ x0 ][ y0 ][ trafoDepth ] ) {	
x1 = x0 + ( ( 1 << log2TrafoSize ) >> 1 )	
y1 = y0 + ( ( 1 << log2TrafoSize ) >> 1 )	
transform_tree( x0, y0, x0, y0, log2TrafoSize - 1, trafoDepth + 1, 0 )	
transform_tree( x1, y0, x0, y0, log2TrafoSize - 1, trafoDepth + 1, 1 )	
transform_tree( x0, y1, x0, y0, log2TrafoSize - 1, trafoDepth + 1, 2 )	
transform_tree( x1, y1, x0, y0, log2TrafoSize - 1, trafoDepth + 1, 3 )	
} else {	
if( PredMode[ x0 ][ y0 ] == MODE_INTRA    trafoDepth != 0    cbf_cb[ x0 ][ y0 ][ trafoDepth ]    cbf_cr[ x0 ][ y0 ][ trafoDepth ] )	
cbf_luma[ x0 ][ y0 ][ trafoDepth ]	ae(v)
transform_unit( x0, y0, xBase, yBase, log2TrafoSize, trafoDepth, blkIdx)	
}	
}	

20

30

40

【 0 0 9 8 】

### 7 . 3 . 1 2 変換単位構文

<code>transform_unit( x0, y0, xBase, yBase, log2TrafoSize, trafoDepth, blkIdx ) {</code>	記述予
<code>if( cbf_luma[ x0 ][ y0 ][ trafoDepth ]    cbf_cb[ x0 ][ y0 ][ trafoDepth ]    cbf_cr[ x0 ][ y0 ][ trafoDepth ] ) {</code>	
<code>if( cu_qp_delta_enabled_flag &amp;&amp; !IsCuQpDeltaCoded ) {</code>	
<code>    cu_qp_delta_abs</code>	ae(v)
<code>    if( cu_qp_delta_abs )</code>	
<code>    cu_qp_delta_sign</code>	ae(v)
<code>    }</code>	
<code>if( cbf_luma[ x0 ][ y0 ][ trafoDepth ] )</code>	
<code>    residual_coding( x0, y0, log2TrafoSize, 0 )</code>	
<code>if( log2TrafoSize &gt; 2 ) {</code>	
<code>    if( cbf_cb[ x0 ][ y0 ][ trafoDepth ] )</code>	
<code>        for ( tIdx = 0; tIdx &lt; TrafoCrCbCnt; tIdx++ ) {</code>	
<code>            residual_coding( x0 + ( ( 1 &lt;&lt; log2CrCbTrafoHorSize ) * (tIdx mod TrafoCrCbHorCnt), y0 + ( 1 &lt;&lt; log2CrCbTrafoVertSize ) * (tIdx div TrafoCrCbVertCnt) ), log2TrafoSize, 1 )</code>	
<code>        }</code>	
<code>    if( cbf_cr[ x0 ][ y0 ][ trafoDepth ] )</code>	
<code>        for ( tIdx = 0; tIdx &lt; TrafoCrCbCnt, tIdx++ ) {</code>	
<code>            residual_coding( x0 + ( ( 1 &lt;&lt; log2CrCbTrafoHorSize ) * (tIdx mod TrafoCrCbHorCnt), y0 + ( 1 &lt;&lt; log2CrCbTrafoVertSize ) * (tIdx div TrafoCrCbVertCnt) ), log2TrafoSize, 2 )</code>	
<code>        }</code>	
<code>    } else if( blkIdx == 3 ) {</code>	
<code>        if( cbf_cb[ xBase ][ yBase ][ trafoDepth ] )</code>	
<code>            residual_coding( xBase, yBase, log2TrafoSize, 1 )</code>	
<code>        if( cbf_cr[ xBase ][ yBase ][ trafoDepth ] )</code>	
<code>            residual_coding( xBase, yBase, log2TrafoSize, 2 )</code>	
<code>        }</code>	
<code>    }</code>	
<code>}</code>	

10

20

## 【 0 0 9 9 】

## 7.4.8.1 概略の符号化単位セマンティックス

変数 `TrafoCrCbHorCnt` および `TrafoCrCbVertCnt` は、以下の通りに導き出される：

30

- ・ `log2TrafoSize` が 5 に等しく、`split_transform_flag` が 0 に等しい場合、`TransformIdxMax` は、以下のように導き出される：
- ・ `chroma_format_idc` が 1 に等しい場合、`TrafoCrCbHorCnt` および `TrafoCrCbVertCnt` は、1 に等しい。
- ・ `chroma_format_idc` が 2 に等しい場合、`TrafoCrCbHorCnt` は、1 に等しく、`TrafoCrCbVertCnt` は、2 に等しい。
- ・ さもなければ、`chroma_format_idc` が 3 に等しい場合、`TrafoCrCbHorCnt` および `TrafoCrCbVertCnt` は、2 に等しい。
- ・ さもなければ、`TrafoCrCbHorCnt` および `TrafoCrCbVertCnt` は、1 に等しい。

40

変数 `TrafoCrCbCnt` は、`TrafoCrCbHorCnt` × `TrafoCrCbVertCnt` として導き出される。

変数 `log2CrCbTrafoHorSize` および `log2CrCbTrafoVertSize` は、以下の通りに導き出される：

- ・ `chroma_format_idc` が 1 に等しい場合、`log2CrCbTrafoHorSize` および `log2CrCbTrafoVertSize` は、`log2TrafoSize - 1` に等しい。
- ・ さもなければ、`chroma_format_idc` が 2 に等しい場合、`log2CrCbTrafoHorSize` は、`log2TrafoSize` に等しく、`log2CrCbTrafoVertSize` は、`min( log2TrafoSize - 1, 4 )` に

50

等しい。

・さもなければ、chroma\_format\_idcが3に等しい場合、log2CrCbTrafoHorSizeおよびlog2CrCbTrafoVertSizeは、min(log2TrafoSize, 4)に等しい。

付録Aの終了

【 0 1 0 0 】

[ 付録 B ]

transform\_tree()を、推測分割から起因する各色差変換に対する色差チャンネルのペアにつき一度起動する。

7.3.1.1 変換ツリー構文

10

transform_tree( x0, y0, xBase, yBase, log2TrafoSize, trafoDepth, blkIdx, chromaOnly ) {	記述子
if( log2TrafoSize <= Log2MaxTrafoSize && log2TrafoSize > Log2MinTrafoSize && trafoDepth < MaxTrafoDepth && !(IntraSplitFlag && trafoDepth == 0) && !chromaOnly )	
split_transform_flag[ x0 ][ y0 ][ trafoDepth ]	ae(v)
if( trafoDepth == 0    log2TrafoSize > 2 ) {	
if( trafoDepth == 0    cbf_cb[ xBase ][ yBase ][ trafoDepth - 1 ] )	
if( TrafoCrCbCnt > 1 ) {	
cbf_cb[ x0 ][ y0 ][ trafoDepth ] = 1	
} else {	
cbf_cb[ x0 ][ y0 ][ trafoDepth ]	ae(v)
}	
if( trafoDepth == 0    cbf_cr[ xBase ][ yBase ][ trafoDepth - 1 ] )	
if( TrafoCrCbCnt > 1 ) {	
cbf_cr[ x0 ][ y0 ][ trafoDepth ] = 1	
} else {	
cbf_cr[ x0 ][ y0 ][ trafoDepth ]	ae(v)
}	
}	
if( split_transform_flag[ x0 ][ y0 ][ trafoDepth ]    TrafoCrCbCnt > 1 ) {	
x1 = x0 + ( ( 1 << log2TrafoSize ) >> 1 )	
y1 = y0 + ( ( 1 << log2TrafoSize ) >> 1 )	
transform_tree( x0, y0, x0, y0, log2TrafoSize - 1, trafoDepth + 1, 0, TrafoCrCbCnt > 1 )	
if( chroma_format_idc != 2 ) {	
transform_tree( x1, y0, x0, y0, log2TrafoSize - 1, trafoDepth + 1, 1, TrafoCrCbCnt > 1 )	
}	
transform_tree( x0, y1, x0, y0, log2TrafoSize - 1, trafoDepth + 1, 2, TrafoCrCbCnt > 1 )	
if( chroma_format_idc != 2 ) {	
transform_tree( x1, y1, x0, y0, log2TrafoSize - 1, trafoDepth + 1, 3, TrafoCrCbCnt > 1 )	
}	
}	
else-if( !split_transform_flag[ x0 ][ y0 ][ trafoDepth ] && TrafoCrCbCnt > 1 ) {	
if( ( PredMode[ x0 ][ y0 ] == MODE_INTRA    trafoDepth != 0    cbf_cb[ x0 ][ y0 ][ trafoDepth ]    cbf_cr[ x0 ][ y0 ][ trafoDepth ] ) && !chromaOnly )	
cbf_luma[ x0 ][ y0 ][ trafoDepth ]	ae(v)
transform_unit( x0, y0, xBase, yBase, log2TrafoSize, trafoDepth, blkIdx, chromaOnly )	
}	
}	

20

30

40

【 0 1 0 1 】

50

## 7.3.1.2 変換単位構文

<code>transform_unit( x0, y0, xBase, yBase, log2TrafoSize, trafoDepth, blkIdx, chromaOnly ) {</code>	記述子
<code>  if( cbf_luma[ x0 ][ y0 ][ trafoDepth ]    cbf_cb[ x0 ][ y0 ][ trafoDepth ]          cbf_cr[ x0 ][ y0 ][ trafoDepth ] ) {</code>	
<code>    if( cu_qp_delta_enabled_flag &amp;&amp; !IsCuQpDeltaCoded        &amp;&amp; !chromaOnly ) {</code>	
<code>      cu_qp_delta_abs</code>	ae(v)
<code>      if( cu_qp_delta_abs )</code>	
<code>      cu_qp_delta_sign</code>	ae(v)
<code>    }</code>	
<code>    if( cbf_luma[ x0 ][ y0 ][ trafoDepth ] )</code>	10
<code>      residual_coding( x0, y0, log2TrafoSize, 0 )</code>	
<code>    if( log2TrafoSize &gt; 2 ) {</code>	
<code>      if( cbf_cb[ x0 ][ y0 ][ trafoDepth ] )</code>	
<code>        residual_coding( x0, y0, log2TrafoSize, 1 )</code>	
<code>      if( cbf_cr[ x0 ][ y0 ][ trafoDepth ] )</code>	
<code>        residual_coding( x0, , log2TrafoSize, 2 )</code>	
<code>    } else if( blkIdx == 3 ) {</code>	
<code>      if( cbf_cb[ xBase ][ yBase ][ trafoDepth ] )</code>	
<code>        residual_coding( xBase, yBase, log2TrafoSize, 1 )</code>	
<code>      if( cbf_cr[ xBase ][ yBase ][ trafoDepth ] )</code>	
<code>        residual_coding( xBase, yBase, log2TrafoSize, 2 )</code>	20
<code>    }</code>	
<code>  }</code>	
<code>}</code>	

## 【 0 1 0 2 】

## 7.4.8.1 概略の符号化単位セマンティックス

変数 `TrafoCrCbHorCnt` および `TrafoCrCbVertCnt` は、以下の通りに導き出される：

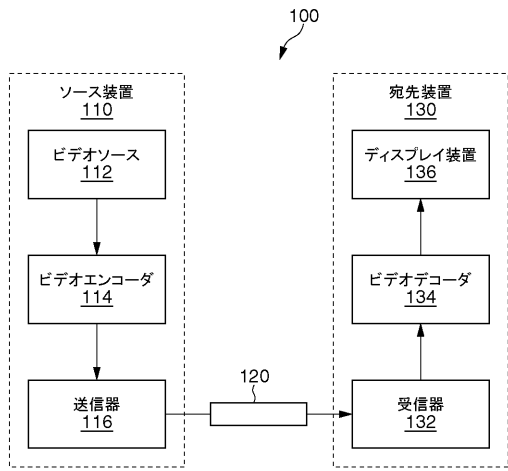
- ・ `log2TrafoSize` が 5 に等しく、`split_transform_flag` が 0 に等しい場合、`TransformIdxMax` は、以下のように導き出される：
- ・ `chroma_format_idc` が 1 に等しい場合、`TrafoCrCbHorCnt` および `TrafoCrCbVertCnt` は、1 に等しい。
- ・ `chroma_format_idc` が 2 に等しい場合、`TrafoCrCbHorCnt` は、1 に等しく、`TrafoCrCbVertCnt` は、2 に等しい。
- ・ 他の場合には、`chroma_format_idc` が 3 に等しい場合、`TrafoCrCbHorCnt` および `TrafoCrCbVertCnt` は、2 に等しい。
- ・ 他の場合には、`TrafoCrCbHorCnt` および `TrafoCrCbVertCnt` は、1 に等しい。

変数 `TrafoCrCbCnt` は、`TrafoCrCbHorCnt × TrafoCrCbVertCnt` として導き出される。

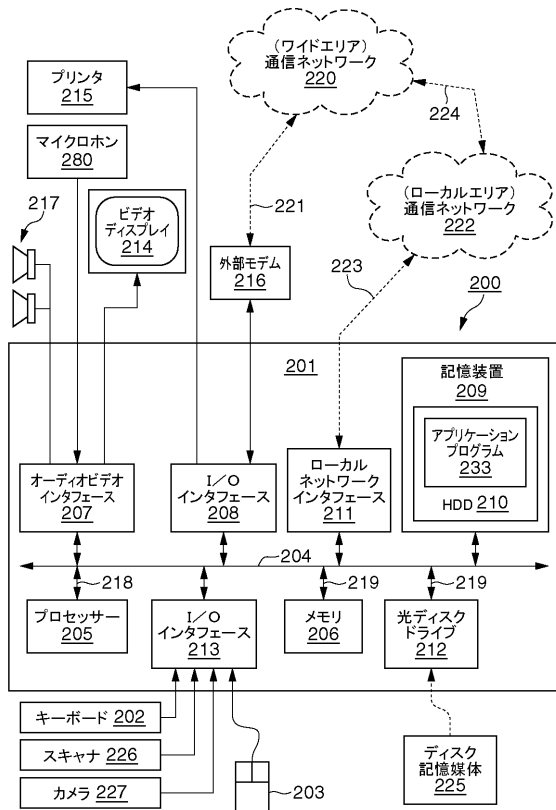
付録 B の終了



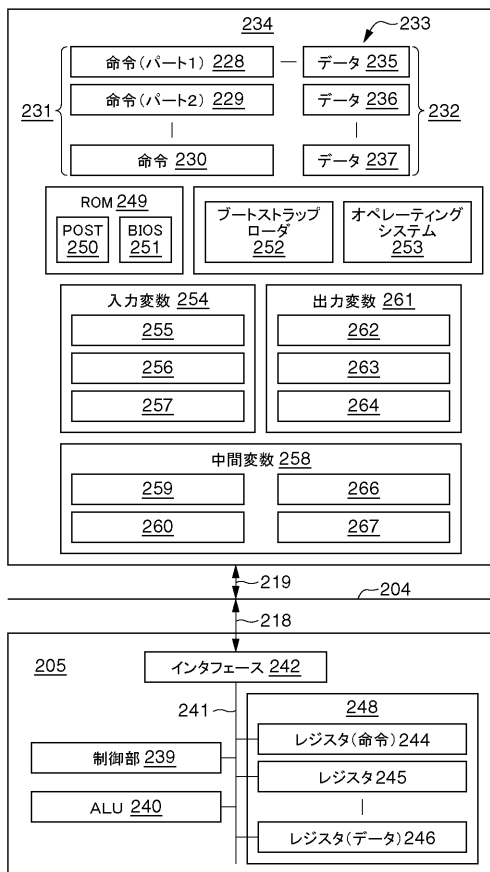
【図1】



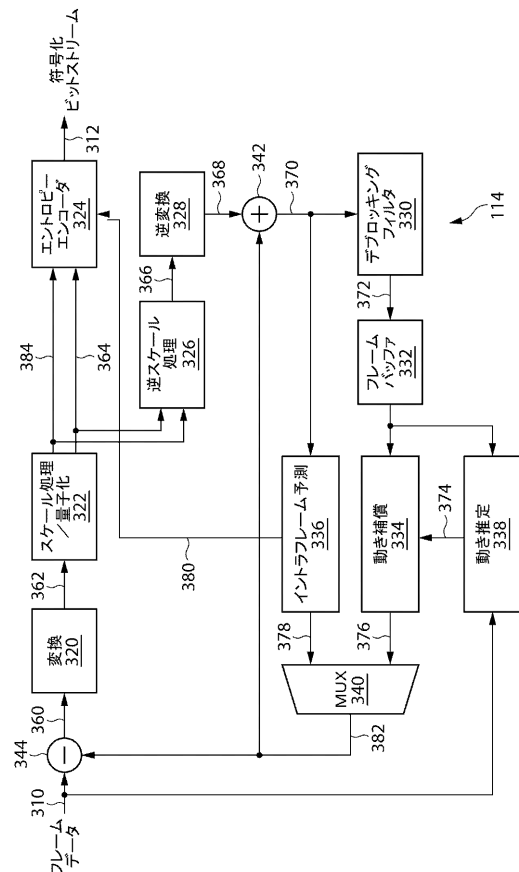
【図2A】



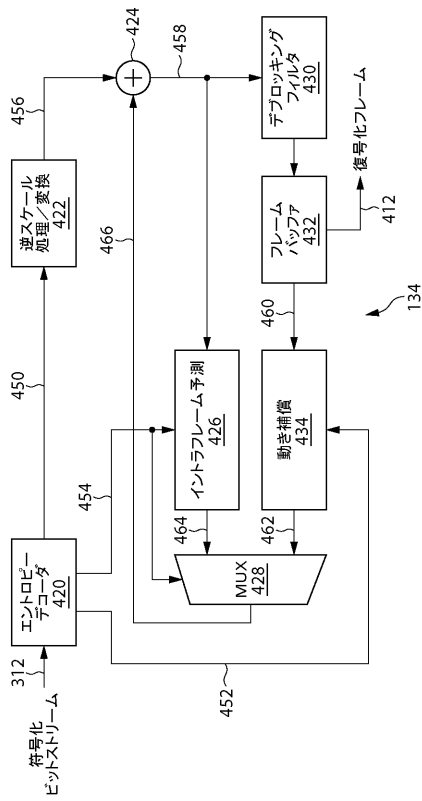
【図2B】



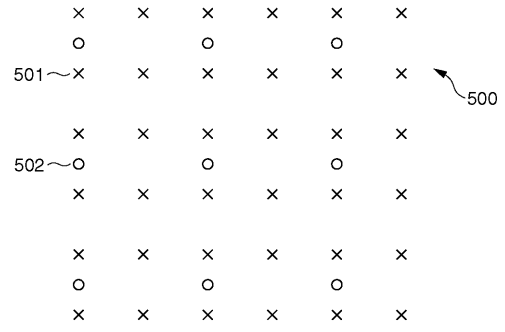
【図3】



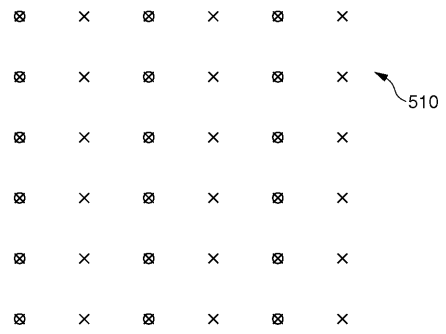
【図4】



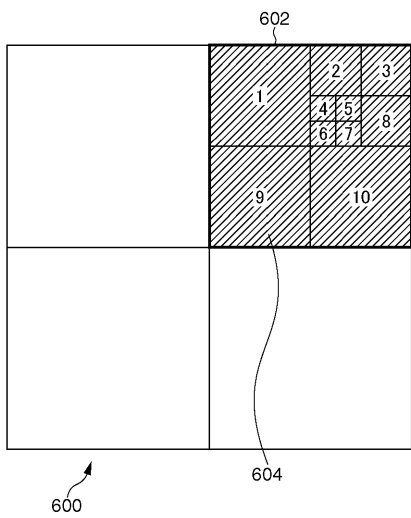
【図5A】



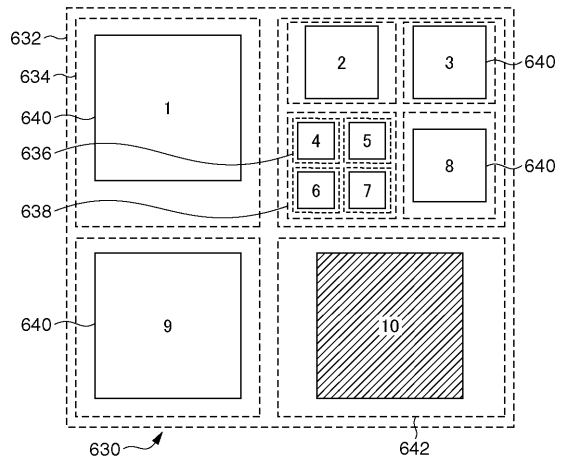
【図5B】



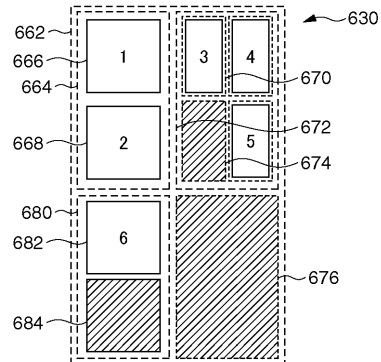
【図6A】



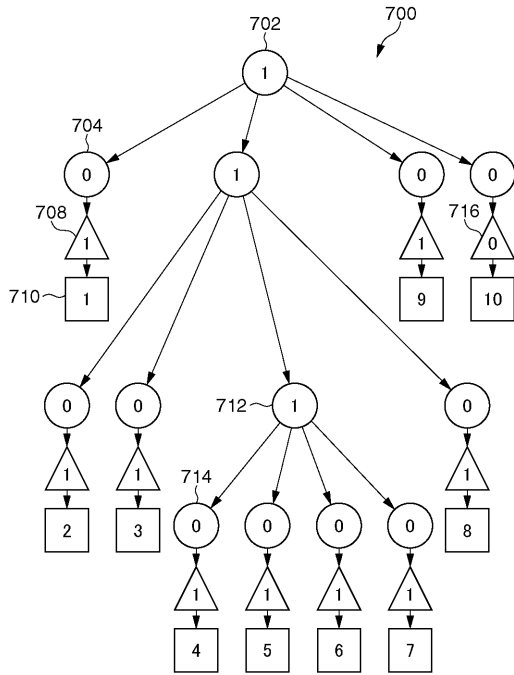
【図6B】



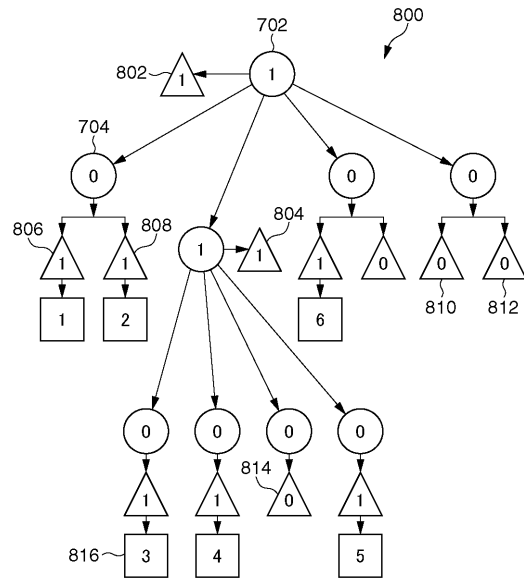
【図6C】



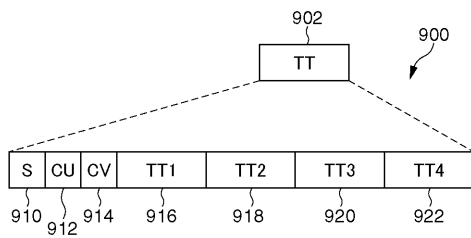
【 図 7 】



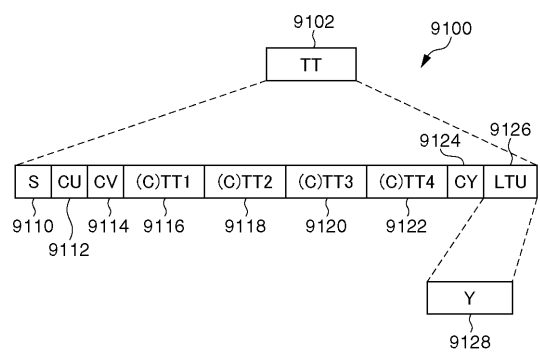
【 図 8 】



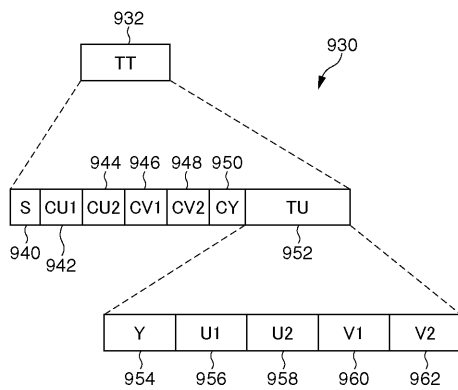
【 図 9 A 】



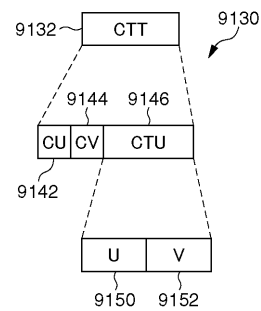
【 図 9 C 】



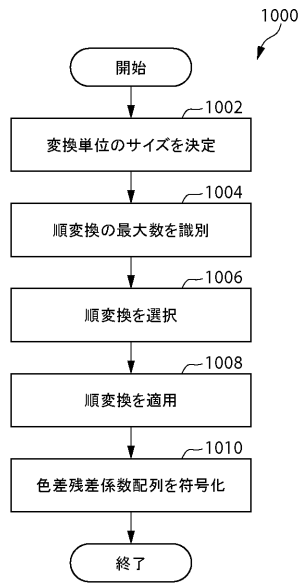
【 図 9 B 】



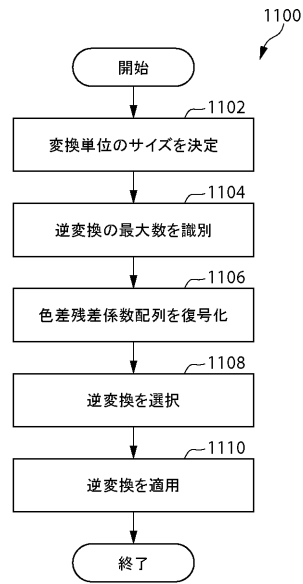
【 図 9 D 】



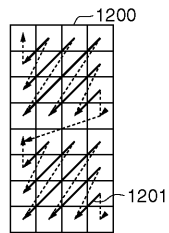
【図10】



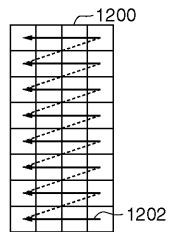
【図11】



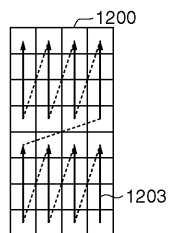
【図12A】



【図12B】



【図12C】



---

フロントページの続き

(72)発明者 ウォロジミール コレスニコフ  
オーストラリア国 2113 ニューサウスウェールズ州 ノースライド、トーマス ホルト ド  
ライブ 1 キヤノン インフォメーション システムズ リサーチ オーストラリア プロプラ  
イェタリー リミテッド内

審査官 堀井 啓明

(56)参考文献 特表2014-535221(JP,A)  
特開2011-077761(JP,A)  
国際公開第2011/087295(WO,A2)

(58)調査した分野(Int.Cl., DB名)  
H04N19/00-19/98