

(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号
特許第7541096号
(P7541096)

(45)発行日 令和6年8月27日(2024.8.27)

(24)登録日 令和6年8月19日(2024.8.19)

(51)国際特許分類

F I

H O 4 N 19/70 (2014.01)

H O 4 N 19/80 (2014.01)

H O 4 N 19/82 (2014.01)

H O 4 N 19/70

H O 4 N 19/80

H O 4 N 19/82

請求項の数 18 (全126頁)

(21)出願番号	特願2022-538826(P2022-538826)	(73)特許権者	504161984
(86)(22)出願日	令和2年12月23日(2020.12.23)		ホアウェイ・テクノロジーズ・カンパニ
(65)公表番号	特表2023-508060(P2023-508060		ー・リミテッド
	A)		中華人民共和国・518129・グァン
(43)公表日	令和5年2月28日(2023.2.28)		ドン・シェンツェン・ロンガン・ディス
(86)国際出願番号	PCT/CN2020/138735		トリクト・バンティアン・(番地なし)
(87)国際公開番号	WO2021/129690		・ホアウェイ・アドミニストレーション
(87)国際公開日	令和3年7月1日(2021.7.1)		・ビルディング
審査請求日	令和4年8月2日(2022.8.2)	(74)代理人	100110364
(31)優先権主張番号	PCT/EP2019/086984		弁理士 実広 信哉
(32)優先日	令和1年12月23日(2019.12.23)	(74)代理人	100133569
(33)優先権主張国・地域又は機関			弁理士 野村 進
	欧州特許庁(EP)	(72)発明者	アナンド・メヘル・コトラ
(31)優先権主張番号	62/960,147		ドイツ・80992・ミュンヘン・リー
(32)優先日	令和2年1月13日(2020.1.13)		スシュトラセ・25・ホアウェイ・テ
	最終頁に続く		最終頁に続く

(54)【発明の名称】 ビデオコーディングのためのクロスコンポーネント適応ループフィルタリング

(57)【特許請求の範囲】

【請求項1】

符号化デバイスによって実施される符号化の方法(4200)であって、

クロスコンポーネント適応ループフィルタ(CC-ALF)を適用してクロマ成分を精緻化するステップ(4201)と、

複数の適応ループフィルタ(ALF)関連構文要素を含むビットストリームを生成するステップ(4202)であって、前記複数のALF関連構文要素がALF関連情報を示す、ステップと

を含み、

前記複数のALF関連構文要素が、シーケンスパラメータセット(SPS)レベル、ピクチャヘッダ、またはスライスヘッダのうちの任意の1つもしくは複数においてシグナリングされ、

前記複数のALF関連構文要素が、

第1の構文要素であって、適応ループフィルタ(ALF)がシーケンスレベルにおいて有効にされているかどうかを示し、前記SPSレベルにおいてシグナリングされる、第1の構文要素と、

第2の構文要素であって、前記クロスコンポーネント適応ループフィルタ(CC-ALF)がシーケンスレベルにおいて有効にされているかどうかを示し、前記SPSレベルにおいてシグナリングされる、第2の構文要素と

を含み、

前記複数のALF関連構文要素は、前記第2の構文要素が、CC-ALFが有効にされていることを示す場合、第4の構文要素を含み、

前記第4の構文要素が、前記ピクチャヘッダにおいてシグナリングされ、

前記第4の構文要素が、Cb色成分に関するCC-ALFが前記ビットストリームに関連付けられたビデオシーケンスの現在のピクチャに対して有効にされているかどうかを示す、方法。

【請求項2】

前記複数のALF関連構文要素は、前記第2の構文要素が、CC-ALFが有効にされていることを示す場合、第3の構文要素を含み、

前記第3の構文要素が、前記ピクチャヘッダにおいてシグナリングされ、

前記第3の構文要素が、CC-ALFが複数のスライスを含む現在のピクチャに対して有効にされているかどうかを示す、請求項1に記載の方法。

10

【請求項3】

前記第4の構文要素が1の値を有する場合、前記Cb色成分に関するCC-ALFが前記現在のピクチャに対して有効にされていることを示し、および/または

前記第4の構文要素が0の値を有する場合、前記Cb色成分に関するCC-ALFが前記現在のピクチャに対して無効にされていることを示す、請求項1または2に記載の方法。

【請求項4】

前記複数のALF関連構文要素は、前記第2の構文要素が、CC-ALFが有効にされていることを示す場合、第7の構文要素を含み、

前記第7の構文要素が、前記ピクチャヘッダにおいてシグナリングされ、

前記第7の構文要素が、Cr色成分に関するCC-ALFが前記ビットストリームに関連付けられたビデオシーケンスの現在のピクチャに対して有効にされているかどうかを指定する、請求項1から3のいずれか一項に記載の方法。

20

【請求項5】

復号デバイスによって実施される復号の方法(4300)であって、

ビットストリームからの複数の適応ループフィルタ(ALF)関連構文要素を解析するステップ(4301)であって、

前記複数のALF関連構文要素が、シーケンスパラメータセット(SPS)レベル、ピクチャヘッダ、またはスライスヘッダのうちの任意の1つもしくは複数から取得され、

前記複数のALF関連構文要素が、

第1の構文要素であって、適応ループフィルタ(ALF)がシーケンスレベルにおいて有効にされているかどうかを示し、前記SPSレベルにおいてシグナリングされる、第1の構文要素と、

30

第2の構文要素であって、クロスコンポーネント適応ループフィルタ(CC-ALF)がシーケンスレベルにおいて有効にされているかどうかを示し、前記SPSレベルにおいてシグナリングされる、第2の構文要素と

を含む、ステップと、

前記複数のALF関連構文要素のうちの少なくとも1つを使用してCC-ALFプロセスを実行するステップ(4302)と

を含み、

前記複数のALF関連構文要素は、前記第2の構文要素が、CC-ALFが有効にされていることを示すものとして取得される場合、第4の構文要素を含み、

40

前記第4の構文要素が、前記ピクチャヘッダから取得され、

前記第4の構文要素が、CC-ALFがビデオシーケンスの現在のピクチャに関するCb色成分に対して有効にされているかどうかを示す、方法。

【請求項6】

前記複数のALF関連構文要素は、前記第2の構文要素が、CC-ALFが有効にされていることを示すものとして取得される場合、第3の構文要素を含み、

前記第3の構文要素が、前記ピクチャヘッダから取得され、

前記第3の構文要素が、CC-ALFが複数のスライスを含む現在のピクチャに対して有効にされているかどうかを示す、請求項5に記載の方法。

50

【請求項 7】

前記第4の構文要素が1の値を有する場合、Cb色成分に関する前記CC-ALFが前記現在のピクチャに対して有効にされていることを示し、および/または

前記第4の構文要素が0の値を有する場合、Cb色成分に関する前記CC-ALFが前記現在のピクチャに対して無効にされていることを示す、請求項5または6に記載の方法。

【請求項 8】

前記複数のALF関連構文要素は、前記第4の構文要素が、前記Cb色成分に関するCC-ALFが前記現在のピクチャに対して有効にされていることを示すものとして取得される場合、第5の構文要素を含み、

前記第5の構文要素が、前記ピクチャヘッダから取得され、

前記第5の構文要素が、前記現在のピクチャ内のすべてのスライスの前記Cb色成分に関連付けられたパラメータセットを示す、請求項5から7のいずれか一項に記載の方法。

【請求項 9】

前記複数のALF関連構文要素は、前記第2の構文要素が、CC-ALFが有効にされていることを示すものとして取得される場合、第7の構文要素を含み、

前記第7の構文要素が、前記ピクチャヘッダから取得され、

前記第7の構文要素が、CC-ALFが前記ビットストリームに関連付けられたビデオシーケンスの現在のピクチャに関するCr色成分に対して有効にされているかどうかを指定する、請求項5から8のいずれか一項に記載の方法。

【請求項 10】

前記第7の構文要素が1の値を有する場合、前記Cr色成分に関するCC-ALFが前記現在のピクチャに対して有効にされていることを示し、および/または

前記第7の構文要素が0の値を有する場合、前記Cr色成分に関するCC-ALFが前記現在のピクチャに対して無効にされていることを示す、請求項9に記載の方法。

【請求項 11】

前記複数のALF関連構文要素は、前記第7の構文要素が、前記Cr色成分に関するCC-ALFが前記現在のピクチャに対して有効にされていることを示すものとして取得される場合、第8の構文要素を含み、

前記第8の構文要素が、前記ピクチャヘッダから取得され、

前記第8の構文要素が、前記現在のピクチャ内のすべてのスライスの前記Cr色成分に関連付けられたパラメータセットを示す、請求項9または10に記載の方法。

【請求項 12】

前記第3の構文要素が、CC-ALFが前記ビットストリームに関連付けられたビデオシーケンスの前記現在のピクチャに対して有効にされていることを示すものとして取得される場合、前記第4の構文要素、第5の構文要素、第7の構文要素、および第8の構文要素が取得され、

前記第5の構文要素が、前記ピクチャヘッダから取得され、前記第5の構文要素が、前記現在のピクチャ内のすべてのスライスの前記Cb色成分に関連付けられたパラメータセットを示し、

前記第7の構文要素が、前記ピクチャヘッダから取得され、前記第7の構文要素が、CC-ALFが前記ビットストリームに関連付けられたビデオシーケンスの現在のピクチャに関するCr色成分に対して有効にされているかどうかを指定し、

前記第8の構文要素が、前記ピクチャヘッダから取得され、前記第8の構文要素が、前記現在のピクチャ内のすべてのスライスの前記Cr色成分に関連付けられたパラメータセットを示す、請求項6に記載の方法。

【請求項 13】

前記複数のALF関連構文要素は、前記第2の構文要素が、CC-ALFが有効にされていることを示すものとして取得された場合、第10の構文要素を含み、

前記第10の構文要素が、スライスヘッダから取得され、

前記第10の構文要素が、Cb色成分に関するCC-ALFが前記ビットストリームに関連付け

10

20

30

40

50

られたビデオシーケンスの現在のピクチャの現在のスライスに対して有効にされているかどうかを示す、請求項5から12のいずれか一項に記載の方法。

【請求項14】

前記第1の構文要素が第1の値を有する場合、前記第2の構文要素が取得され、または前記第2の構文要素が、少なくとも前記第1の構文要素の値に基づいて条件付きで取得される、請求項5から13のいずれか一項に記載の方法。

【請求項15】

ビデオデータを符号化するためのデバイスであって、
ビデオデータメモリと、
ビデオエンコーダと
を備え、
前記ビデオエンコーダが、
クロスコンポーネント適応ループフィルタ(CC-ALF)を適用してクロマ成分を精緻化することと、

複数の適応ループフィルタ(ALF)関連構文要素を含むビットストリームを生成することであって、前記複数のALF関連構文要素が、ALF関連情報を示す、ことと、

を行うように構成され、

前記複数のALF関連構文要素が、シーケンスパラメータセット(SPS)レベル、ピクチャヘッダ、またはスライスヘッダのうちの任意の1つまたは複数においてシグナリングされ、

前記複数のALF関連構文要素が、

第1の構文要素であって、適応ループフィルタ(ALF)がシーケンスレベルにおいて有効にされているかどうかを示し、前記SPSレベルにおいてシグナリングされる、第1の構文要素と、

第2の構文要素であって、前記クロスコンポーネント適応ループフィルタ(CC-ALF)がシーケンスレベルにおいて有効にされているかどうかを示し、前記SPSレベルにおいてシグナリングされる、第2の構文要素と

を含み、

前記複数のALF関連構文要素は、前記第2の構文要素が、CC-ALFが有効にされていることを示す場合、第4の構文要素を含み、

前記第4の構文要素が、前記ピクチャヘッダにおいてシグナリングされ、

前記第4の構文要素が、Cb色成分に関するCC-ALFが前記ビットストリームに関連付けられたビデオシーケンスの現在のピクチャに対して有効にされているかどうかを示す、デバイス。

【請求項16】

ビデオデータを復号するためのデバイスであって、
ビデオデータメモリと、
ビデオデコーダと
を備え、

前記ビデオデコーダが、

ビットストリームからの複数の適応ループフィルタ(ALF)関連構文要素を解析することであって、

前記複数のALF関連構文要素が、シーケンスパラメータセット(SPS)レベル、ピクチャヘッダ、またはスライスヘッダのうちの任意の1つまたは複数から取得され、

前記複数のALF関連構文要素が、

第1の構文要素であって、適応ループフィルタ(ALF)がシーケンスレベルにおいて有効にされているかどうかを示し、前記SPSレベルにおいてシグナリングされる、第1の構文要素と、

第2の構文要素であって、クロスコンポーネント適応ループフィルタ(CC-ALF)がシーケンスレベルにおいて有効にされているかどうかを示し、前記SPSレベルにおいてシグナリングされる、第2の構文要素と

10

20

30

40

50

を含む、ことと、

前記複数のALF関連構文要素のうちの少なくとも1つを使用してCC-ALFプロセスを実行することと

を行うように構成され、

前記複数のALF関連構文要素は、前記第2の構文要素が、CC-ALFが有効にされていることを示すものとして取得される場合、第4の構文要素を含み、

前記第4の構文要素が、前記ピクチャヘッダから取得され、

前記第4の構文要素が、CC-ALFがビデオシーケンスの現在のピクチャに関するCb色成分に対して有効にされているかどうかを示す、デバイス。

【請求項17】

コンピューティングデバイスによって実行されると、前記コンピューティングデバイスに請求項1から4のいずれか一項に記載の方法を実行させるコンピュータ実行可能命令を記憶した、コンピュータ可読記憶媒体。

【請求項18】

コンピューティングデバイスによって実行されると、前記コンピューティングデバイスに請求項5から14のいずれか一項に記載の方法を実行させるコンピュータ実行可能命令を記憶した、コンピュータ可読記憶媒体。

【発明の詳細な説明】

【技術分野】

【0001】

本出願(開示)の実施形態は、概して、ピクチャ処理の分野に関し、より詳細には、ループ内フィルタまたはポストループフィルタとしてのクロスコンポーネント適応ループフィルタ(CC-ALF(cross-component adaptive loop filter))、およびクロスコンポーネントALF(CCALF)のための高レベル構文に関する。

【背景技術】

【0002】

画像コーディング(符号化および復号)は、多種多様なデジタル画像アプリケーション、たとえば、ブロードキャストデジタルTV、インターネットおよびモバイルネットワークを介するビデオ伝送、ビデオチャットなどのリアルタイム会話アプリケーション、ビデオ会議、DVDおよびBlu-ray(登録商標)ディスク、ビデオコンテンツ取得および編集システム、ならびにセキュリティアプリケーションのカムコーダにおいて使用される。

【0003】

1990年におけるH.261規格におけるブロックベースのハイブリッドビデオコーディング方法が開発されて以来、新しいビデオコーディング技法およびツールが開発され、新しいビデオコーディング規格のための基礎を形成した。大部分のビデオコーディング規格の目標の1つは、ピクチャ品質を犠牲にすることなく、以前の規格と比較してビットレートの低減を達成することであった。さらなるビデオコーディング規格は、MPEG-1ビデオ、MPEG-2ビデオ、ITU-T H.262/MPEG-2、ITU-T H.263、ITU-T H.264/MPEG-4、Part 10、アドバンスドビデオコーディング(AVC(Advanced Video Coding))、ITU-T H.265、高効率ビデオコーディング(HEVC(High Efficiency Video Coding))、ITU-T H.266/多用途ビデオコーディング(VVC(Versatile video coding))、およびそれらの規格の拡張、たとえば、スケーラビリティおよび/または3次元(3D)拡張を含む。

【0004】

ブロックベースの画像コーディング方式は、ブロックのエッジに沿ってエッジアーティファクトが現れる可能性があるという共通点を有する。これらのアーティファクトは、コーディングブロックの独立したコーディングに起因する。これらのエッジアーティファクトは、しばしば、ユーザに対して容易に可視になる。ブロックベースの画像コーディングにおける目標は、エッジアーティファクトを可視性しきい値未満に低減することである。これは、デブロッキングフィルタ、SAO、および適応ループフィルタ(ALF)などのループフィルタリングを実行することによって行われる。フィルタリングプロセスの順序は、デ

10

20

30

40

50

ブロッキングフィルタ、次いでSAO、次いでALFである。さらに、クロスコンポーネント適応ループフィルタ(CC-ALF)がさらに使用される。

【0005】

特に、クロスコンポーネント適応ループフィルタ(CC-ALF)について、各クロマ成分を精緻化するために、ルマサンプル値が使用される。処理は、Cb成分とCr成分の両方において行われる必要があり、クロスコンポーネント適応ループフィルタリングは、計算上複雑になる可能性があり、したがって、特にハードウェア実装の場合、追加のパイプライン待ち時間を追加する場合がある。

【発明の概要】

【課題を解決するための手段】

10

【0006】

上記の課題に鑑み、本開示は、クロスコンポーネント適応ループフィルタリングと、CC-ALFのための構文要素とを改善することを目的とする。本開示は、とりわけ、シグナリングオーバーヘッドを低減してクロスコンポーネント適応ループフィルタリングを実行することができる装置、エンコーダ、デコーダ、および対応する方法を提供する目的に関係し、具体的には、(ビット数に関して)スライスヘッダのオーバーヘッドは、低減され得、したがって、フィルタリングは、より効率的であり得る。

【0007】

本開示の例は、コーディング性能を改善することができ、それによって、ビデオ信号のコーディング効率を改善することができる、画像を符号化および復号するための装置および方法を提供する。本開示は、このファイル内に含まれる例および特許請求の範囲において詳述されている。

20

【0008】

本出願の実施形態は、独立請求項に従って符号化および復号のための装置および方法を提供し、したがって、クロスコンポーネントALFの複雑さは、低減され得、ループ内フィルタまたはポストループフィルタとしてのクロスコンポーネント適応ループフィルタ(CC-ALF)の性能は、それぞれ改善され得る。

【0009】

前述の目的および他の目的は、独立請求項の主題によって達成され得る。さらなる実装形態は、従属請求項、説明、および図から明らかである。

30

【0010】

特定の実施形態は、添付の独立請求項において概説され、他の実施形態は、従属請求項において概説される。

【0011】

第1の態様によれば、本開示は、

クロスコンポーネント適応ループフィルタ(CC-ALF)を適用することによってフィルタリングプロセス(クロスコンポーネントフィルタリングプロセスなど)を実行するステップと、複数のCC-ALF関連構文要素(M個のCC-ALF関連構文要素などであり、 $M \geq 1$ 、Mは整数)を含むビットストリームを生成するステップであって、複数のCC-ALF関連構文要素が、CC-ALF関連情報を示す、ステップと

40

を含む、符号化デバイスによって実施される符号化の方法に関連し、

複数のCC-ALF関連構文要素が、ビデオパラメータセット(VPS(video parameter set))レベル、シーケンスパラメータセット(PPS(sequence parameter set))レベル、ピクチャパラメータセット(PPS(picture parameter set))レベル、ピクチャヘッダ、スライスヘッダ、もしくはタイルヘッダのうちの任意の1つまたは複数においてシグナリングされるか、または

複数のCC-ALF関連構文要素が、シーケンスパラメータセット(PPS)レベルおよび/もしくはピクチャヘッダにおいてシグナリングされる。

【0012】

このビットストリームは、本出願が実際に適用されるレベルまたは本出願が関係するレ

50

ベルに関するビットストリームの構造において関連情報を提供しながら、サイズにおいて低減され得、シグナリングオーバーヘッドを低減してクロスコンポーネント適応ループフィルタリングを実行することを可能にし、したがって、フィルタリングは、より効率的になり得、コーディング効率の改善が達成される。

【0013】

第2の態様によれば、本開示は、

ビデオ信号のビットストリームから1つまたは複数の構文要素を解析するステップであって、構文要素が、クロスコンポーネント適応ループフィルタ(CC-ALF)関連情報を示し、構文要素が、ビットストリームのビデオパラメータセット(VPS)レベル、シーケンスパラメータセット(SPS)レベル、ピクチャパラメータセット(PPS)レベル、ピクチャヘッダ、スライスヘッダ、もしくはタイルヘッダのうちの1つまたは複数から取得されるか、または構文要素が、シーケンスパラメータセット(SPS)レベルおよび/またはピクチャヘッダから取得される、ステップと、

構文要素に基づくか、または構文要素の値に基づいてCC-ALFを適用することによって、フィルタリングプロセス(クロスコンポーネントフィルタリングプロセスなど)を実行するステップと

を含む、復号デバイスによって実施される復号の方法に関する。

【0014】

この方法は、復号中にビットストリームから関連情報を取得することを可能にし得、ビットストリームが、サイズにおいて低減し、データの改善された圧縮を可能にし、また、この方法は、シグナリングオーバーヘッドを低減してクロスコンポーネント適応ループフィルタリングを実行することを可能にし、したがって、フィルタリングは、より効率的になり得、コーディング効率の改善が達成される。

【0015】

第3の態様によれば、本発明は、ビデオデータを復号する装置に関する。装置は、本開示の第1の態様による方法を実行するための処理回路を備える。

【0016】

第4の態様によれば、本発明は、ビデオデータを符号化するための装置に関する。装置は、本開示の第2の態様による方法を実行する処理回路を備える。

【0017】

本開示の第1の態様による方法は、本開示の第3の態様による装置によって実行され得る。さらに、本開示の第3の態様による方法のさらなる特徴および実装形態は、本開示の第1の態様による装置の特徴および実装形態に対応する。

【0018】

本開示の第2の態様による方法は、本発明の第4の態様による装置によって実行され得る。本開示の第4の態様による方法のさらなる特徴および実装形態は、本開示の第2の態様による装置の特徴および実装形態に対応する。

【0019】

第5の態様によれば、本開示は、ビデオストリームを復号するための装置に関し、プロセッサとメモリとを含む。メモリは、プロセッサに第1の態様による方法を実行させる命令を記憶している。

【0020】

第6の態様によれば、本開示は、ビデオストリームを符号化するための装置に関し、プロセッサとメモリとを含む。メモリは、プロセッサに第2の態様による方法を実行させる命令を記憶している。

【0021】

第7の態様によれば、実行されると、ビデオデータをコーディングするように構成された1つまたは複数のプロセッサをもたらし命令を記憶したコンピュータ可読記憶媒体が提案される。命令は、1つまたは複数のプロセッサに、第1もしくは第2の態様または第1もしくは第2の態様の任意の可能な実施形態による方法を実行させる。

10

20

30

40

50

【0022】

第8の態様によれば、本開示は、コンピュータ上で実行されると、第1もしくは第2の態様または第1もしくは第2の態様の任意の可能な実施形態による方法を実行するためのプログラムコードを含むコンピュータプログラムに関する。

【0023】

本開示は、符号化デバイスによって実施される符号化の方法を提供し、方法は、クロスコンポーネント適応ループフィルタCC-ALFを適用してクロマ成分を精緻化するステップと、

複数のALF関連構文要素(以下で使用されるCC-ALF関連構文要素)を含むビットストリームを生成するステップであって、複数のCC-ALF関連構文要素が、ALF関連情報(以下で使
用されるCC-ALF関連情報)を示す、ステップと
を含み、

複数のCC-ALF関連構文要素が、シーケンスパラメータセット(SPS)レベル、ピクチャヘッダ、またはスライスヘッダのうちの任意の1つまたは複数においてシグナリングされ、

複数のCC-ALF関連構文要素が、第1の構文要素であって、クロスコンポーネント適応ループフィルタを含む適応ループフィルタ(ALF)がシーケンスレベルにおいて有効にされているかどうかを示し、SPSレベルにおいてシグナリングされる、第1の構文要素と、第2の構文要素であって、クロスコンポーネント適応ループフィルタがシーケンスレベルにおいて有効にされているかどうかを示し、SPSレベルにおいてシグナリングされる、第2の構文要素とを含む。

【0024】

本明細書では、概して、第1および第2の構文要素がSPSレベルにおいてシグナリングされることが指定されているが、これは、少なくとも第2の構文要素のシグナリングが無条件であることを意味していない。むしろ、この実施形態は、第2の構文要素が、たとえば、第1の構文要素の値に基づいて、および/または以下でさらに論じるように別の構文要素の値に依存してシグナリングされる実現例も包含する。

【0025】

第1の構文要素は、以下、sps_alf_enabled_flagと呼ばれる場合があり、第2の構文要素は、sps_ccalf_enabled_flagと呼ばれる場合がある。しかしながら、これは、本明細書で使用される命名に過ぎない。本発明は、本明細書で言及される第1もしくは第2の構文要素または任意の他の構文要素の特定の名前に限定されない。

【0026】

CC-ALFは、ALFの特殊な種類であり、CC-ALFは、ALFが有効にされているかどうかに依存する場合があり、したがって、第1の構文要素、すなわち、sps_alf_enabled_flagは、CC-ALFにも関連し、したがって、CC-ALF関連の構文要素が以下で使用される場合があることが理解され得る。同様に、第1の構文要素、すなわち、sps_alf_enabled_flagによって示される情報も、CC-ALFに関連するので、CC-ALF関連情報が以下で使用される場合がある。

【0027】

これによって、たとえば、シーケンス内のすべてのピクチャに対して使用され得る情報は、SPSレベルにおいて効率的にシグナリングされ得、冗長な情報の量を低減することによってビットストリームのサイズを低減し、シグナリングオーバーヘッドを低減してクロスコンポーネント適応ループフィルタリングを実行することを可能にし、したがって、フィルタリングは、より効率的になり得、コーディング効率の改善が達成される。

【0028】

一実施形態において、複数のCC-ALF関連構文要素は、第2の構文要素が、CC-ALFが有効にされていることを示す場合、第3の構文要素を含み、第3の構文要素は、ピクチャヘッダにおいてシグナリングされ、第3の構文要素は、CC-ALFが複数のスライスを含む現在のピクチャに対して有効にされているかどうかを示す。

【0029】

この第3の構文要素は、pic_ccalf_enabled_flagと呼ばれる場合もある。この実施形態では、ビットストリームのサイズを小さく保ちながら、完全なピクチャに関する関連CC-ALF情報がシグナリングされ得る。一例において、スライス1...スライスNは、同じCCALF情報を共有し、したがって、共通情報は、各スライスヘッダがそれらを冗長に送信する代わりに、ピクチャヘッダから直接継承され得る。

【0030】

さらなる実施形態において、複数のCC-ALF関連構文要素は、第2の構文要素が、CC-ALFが有効にされていることを示す場合、第4構文要素を含み、第4の構文要素は、ピクチャヘッダにおいてシグナリングされ、第4の構文要素は、Cb色成分に関するCC-ALFがビットストリームに関連付けられたビデオシーケンスの現在のピクチャに対して有効にされているかどうかを示す。

10

【0031】

第4の構文要素は、たとえば、pic_cross_component_alf_cb_enabled_flagで示され得る。しかしながら、これは、必須ではない。第4の構文要素は、たとえば、スライスヘッダ内の冗長な情報の量を小さく保ちながら、Cb色成分に関するCC-ALFを効率的にシグナリングすることができる。

【0032】

たとえば、第4の構文要素が1の値を有する場合、それは、Cb色成分に関するCC-ALFが現在のピクチャに対して有効にされていることを示し、および/または第4の構文要素が0の値を有する場合、それは、Cb色成分に関するCC-ALFが現在のピクチャに対して無効にされていることを示すことがさらに規定され得る。

20

【0033】

一実施形態において、複数のCC-ALF関連構文要素は、第4の構文要素が、Cb色成分に関するCC-ALFが現在のピクチャに対して有効にされていることを示す場合、第5の構文要素を含み、第5の構文要素は、ピクチャヘッダにおいてシグナリングされ、第5の構文要素は、現在のピクチャ内のすべてのスライスのCb色成分が参照するパラメータセットを示す。この構文要素は、pic_cross_component_alf_cb_aps_idで示され得る。しかしながら、これは、単なる命名であり、本開示を限定するように解釈されない。この実施形態は、ピクチャレベルにおいてすでにピクチャのすべてのスライスについて提供されているパラメータセットをシグナリングすることを可能にし得、それによって、冗長情報の量を低減する。

30

【0034】

複数のCC-ALF関連構文要素は、第2の構文要素が、CC-ALFが有効にされていることを示す場合、第7の構文要素を含み、第7の構文要素は、ピクチャヘッダにおいてシグナリングされ、第7の構文要素は、Cr色成分に関するCC-ALFがビットストリームに関連付けられたビデオシーケンスの現在のピクチャに対して有効にされているかどうかを指定することがさらに規定され得る。この構文要素は、たとえば、本開示を限定することなく、pic_cross_component_alf_cr_enabled_flagで示され得る。この構文要素を用いて、Cr色成分に対するCC-ALFの有効化が、確実にシグナリングされ得る。

【0035】

40

第7の構文要素が1の値を有する場合、それは、Cr色成分に関するCC-ALFが現在のピクチャに対して有効にされていることを示し、および/または第7の構文要素が0の値を有する場合、それは、Cr色成分に関するCC-ALFが現在のピクチャに対して無効にされていることを示すことがさらに規定され得る。

【0036】

一実施形態において、複数のCC-ALF関連構文要素は、第7の構文要素が、Cr色成分に関するCC-ALFが現在のピクチャに対して有効にされていることを示す場合、第8の構文要素を含み、第8の構文要素は、ピクチャヘッダにおいてシグナリングされ、第8の構文要素は、現在のピクチャ内のすべてのスライスのCr色成分に関連付けられたパラメータセットを示す。第8の構文要素は、pic_cross_component_alf_cr_aps_idで示され得るが、これ

50

は、単なる一例である。この構文要素は、フィルタリング中に使用され得る関連するパラメータに関する情報を提供することができる。

【0037】

第4の構文要素、第5の構文要素、第6の構文要素、第7の構文要素、第8の構文要素、および第9の構文要素は、第3の構文要素が、CC-ALFがビットストリームに関連付けられたビデオシーケンスの現在のピクチャに対して有効にされていることを示す場合、シグナリングされることがさらに規定され得る。CC-ALFが無効にされている場合、これらの要素は、デフォルト値に設定され、依然としてビットストリームにおいてシグナリングされ得る。代替案において、CC-ALFが無効にされている場合、これらの構文要素は、ビットストリームにおいてシグナリングされなくてよく、それによって、使用されない情報がビットストリームから排除されるので、そのサイズを低減する。

10

【0038】

さらなる実施形態において、複数のCC-ALF関連構文要素は、第2の構文要素が、CC-ALFが有効にされていることを示す場合、第10の構文要素を含み、第10の構文要素は、スライスヘッダにおいてシグナリングされ、第10の構文要素は、Cb色成分に関するCCALFがビットストリームに関連付けられたビデオシーケンスの現在のピクチャの現在のスライスに対して有効にされているかどうかを示す。この構文要素は、これが本開示を限定することを意図されることなく、`slice_cross_component_alf_cb_enabled_flag`と呼ばれる場合がある。これによって、CC-ALFがCb色成分に対して有効にされるべきかどうかの効率的なシグナリングが提供され得る。

20

【0039】

第10の構文要素が1の値を有する場合、それは、Cb色成分に関するCCALFが現在のスライスに対して有効にされていることを示し、および/または第10の構文要素が0の値を有する場合、それは、Cb色成分に関するCCALFが現在のスライスに対して無効にされていることを示すことがさらに規定され得る。

【0040】

さらなる実施形態において、複数のCC-ALF関連構文要素は、第10の構文要素が、Cb色成分に関するCC-ALFが現在のスライスに対して有効にされている場合、第11の構文要素を含み、第10の構文要素は、スライスヘッダにおいてシグナリングされ、第10の構文要素は、現在のスライスのCb色成分が参照するパラメータセットを指定する。この構文要素は、`pic_cross_component_alf_cb_aps_id`で示され得るが、これは、本開示を限定することを意図していない。

30

【0041】

さらなる実施形態において、複数のCC-ALF関連構文要素は、第2の構文要素が、CC-ALFが有効にされていることを示す場合、第12の構文要素を含み、第12の構文要素は、スライスヘッダにおいてシグナリングされ、第12の構文要素は、Cr色成分に関するCCALFがビットストリームに関連付けられたビデオシーケンスの現在のピクチャの現在のスライスに対して有効にされているかどうかを示す。第12の構文要素は、たとえば、`slice_cross_component_alf_cr_enabled_flag`で示され得る。

【0042】

さらなる実施形態において、第12の構文要素が1の値を有する場合、それは、Cr色成分に関するCCALFが現在のスライスに対して有効にされていることを示し、および/または第12の構文要素が0の値に設定されている場合、それは、Cr色成分に関するCCALFが現在のスライスに対して無効にされていることを示すことが規定される。

40

【0043】

複数のCC-ALF関連構文要素は、第12の構文要素が、Cr色成分に関するCC-ALFが現在のスライスに対して有効にされていることを示す場合、第13の構文要素を含み、第13の構文要素は、スライスヘッダにおいてシグナリングされ、第13の構文要素は、現在のスライスのCr色成分が参照するパラメータセットを指定することもさらに規定され得る。第13の構文要素は、たとえば、`slice_cross_component_alf_cr_aps_id`で示され得る。これ

50

は、Cr色成分に関連付けられたフィルタリングについて使用されるべきパラメータに関する情報を効率的に提供することができる。

【0044】

第2の構文要素は、第1の構文要素が第1の値を有する場合にシグナリングされるか、または第2の構文要素は、少なくとも第1の構文要素の値に基づいて条件付きでシグナリングされることが規定され得る。ALFが有効にされていない場合(第1の構文要素によってシグナリングされる)、CC-ALFも有効にされない。この場合、第2の構文要素を提供しないことによって、ビットストリームのサイズは、さらに低減され得る。

【0045】

一実施形態において、複数のCC-ALF関連構文要素は、SPSレベルにおいてシグナリングされる第14の構文要素を含み、第14の構文要素は、CC-ALFへの入力の種類を示す。第14の構文要素は、ChromaArrayTypeで示され得るが、これは、本開示を限定していない。

10

【0046】

より具体的には、第2の構文要素は、第1の構文要素が第1の値を有し、かつ第14の構文要素が第2の値を有する場合にシグナリングされる。第2の値は、CC-ALFが有効にされるべきではないことを示す特定の値とは異なる任意の値であり得る。

【0047】

さらなる特定の実施形態において、第2の構文要素は、第1の構文要素が1に等しい値を有し、かつ第14の構文要素が0に等しくない値を有する場合にシグナリングされる。

20

【0048】

上記の実施形態のいずれかについて、CC-ALFが、適応ループフィルタ処理の一部として動作し、少なくとも1つのクロマ成分を精緻化するためにルマサンプル値を利用することが規定され得る。

【0049】

本開示は、復号デバイスによって実施される復号の方法をさらに提供し、方法は、ビットストリームからの複数のクロスコンポーネント適応ループフィルタCC-ALF関連構文要素を解析するステップであって、複数の構文要素、複数のCC-ALF関連構文要素が、シーケンスパラメータセット(SPS)レベル、ピクチャヘッダ、またはスライスヘッダのうちの任意の1つまたは複数から取得され、

30

複数のCC-ALF関連構文要素が、第1の構文要素であって、クロスコンポーネント適応ループフィルタを含む適応ループフィルタ(ALF)がシーケンスレベルにおいて有効にされているかどうかを示し、SPSレベルにおいてシグナリングされる、第1の構文要素と、第2の構文要素であって、クロスコンポーネント適応ループフィルタがシーケンスレベルにおいて有効にされているかどうかを示し、第2の構文要素がSPSレベルにおいてシグナリングされる、第2の構文要素とを含む、ステップと、

複数のCC-ALF関連構文要素のうちの少なくとも1つを使用してCC-ALFプロセスを実行するステップと

を含む。

【0050】

40

本明細書では、概して、第1および第2の構文要素がSPSレベルにおいて提供されることが指定されているが、これは、少なくとも第2の構文要素が無条件な方法でビットストリームにおいて提供されることを意味していない。むしろ、この実施形態は、たとえば、第1の構文要素の値に基づいて、および/または以下でさらに論じるように別の構文要素の値に依存して、第2の構文要素がビットストリームの一部である実現例も包含する。

【0051】

第1の構文要素は、以下ではsps_alf_enabled_flagと呼ばれる場合があり、第2の構文要素は、sps_ccalf_enabled_flagと呼ばれる場合がある。しかしながら、これは、本明細書で使用される命名に過ぎない。本発明は、本明細書で言及される第1もしくは第2の構文要素または任意の他の構文要素の特定の名前に限定されない。

50

【0052】

これによって、たとえば、シーケンス内のすべてのピクチャに対して使用され得る情報は、SPSレベルにおいて効率的にデコーダに提供され得、冗長な情報の量を低減することによってビットストリームのサイズを低減し、シグナリングオーバーヘッドを低減してクロスコンポーネント適応ループフィルタリングを実行することを可能にし、したがって、フィルタリングは、より効率的になり得、コーディング効率の改善が達成される。

【0053】

複数のCC-ALF関連構文要素は、第2の構文要素が、CC-ALFが有効にされていることを示すものとして取得される場合、第3の構文要素を含み、第3の構文要素は、ピクチャヘッダから取得され、第3の構文要素は、CC-ALFが複数のスライスを含む現在のピクチャに対して有効にされているかどうかを示すことが規定され得る。この第3の構文要素は、pic_cross_component_alf_enabled_flagと呼ばれる場合もある。この実施形態では、ビットストリームのサイズを小さく保ちながら、完全なピクチャに関係する関連CC-ALF情報が提供され得る。

10

【0054】

さらなる実施形態において、複数のCC-ALF関連構文要素は、第2の構文要素が、CC-ALFが有効にされていることを示すものとして取得される場合、第4の構文要素を含み、第4の構文要素が、ピクチャヘッダから取得され、第4の構文要素が、CC-ALFがビデオシーケンスの現在のピクチャに関するCb色成分に対して有効にされているかどうかを示す。第4の構文要素は、たとえば、第4の構文要素は、pic_cross_component_alf_cb_enabled_flagで示され得る。しかしながら、これは、必須ではない。第4の構文要素は、たとえば、スライスヘッダ内の冗長な情報の量を小さく保ちながら、デコーダにCb色成分に関するCC-ALFを効率的に有効にさせることができる。

20

【0055】

さらなる実施形態において、第4の構文要素が1の値を有する場合、それは、Cb色成分に関するCC-ALFが現在のピクチャに対して有効にされていることを示し、および/または第4の構文要素が0の値を有する場合、それは、Cb色成分に関するCC-ALFが現在のピクチャに対して無効にされていることを示すことが規定され得る。

【0056】

さらに、複数のCC-ALF関連構文要素は、第4の構文要素が、Cb色成分に関するCC-ALFが現在のピクチャに対して有効にされていることを示すものとして取得される場合、第5の構文要素を含み得、第5の構文要素は、ピクチャヘッダから取得され、第5の構文要素は、現在のピクチャ内のすべてのスライスのCb色成分に関連付けられたパラメータセットを示す。この構文要素は、pic_cross_component_alf_cb_aps_idで示され得る。しかしながら、これは、単なる命名であり、本開示を限定するように解釈されない。この実施形態は、ピクチャレベルにおいてすでにピクチャのすべてのスライスについて提供されているパラメータセットを提供することを可能にし得、それによって、冗長情報の量を低減する。

30

【0057】

さらなる実施形態において、複数のCC-ALF関連構文要素は、第2の構文要素が、CC-ALFが有効にされていることを示すものとして取得される場合、第7の構文要素を含み、第7の構文要素は、ピクチャヘッダから取得され、第7の構文要素は、CC-ALFがビットストリームに関連付けられたビデオシーケンスの現在のピクチャに関するCr色成分に対して可能にされているかどうかを指定する。この構文要素は、たとえば、本開示を限定することなく、pic_cross_component_alf_cr_enabled_flagで示され得る。この構文要素では、この構文要素CC-ALFを用いて、デコーダは、CC-ALFがCr色成分に対して有効にされているかどうかを信頼性高く判断することができる。

40

【0058】

第7の構文要素が1の値を有する場合、それは、Cr色成分に関するCC-ALFが現在のピクチャに対して有効にされていることを示し、および/または第7の構文要素が0の値を有する場合、それは、Cr色成分に関するCC-ALFが現在のピクチャに対して無効にされている

50

ことを示すことが規定され得る。

【0059】

さらなる実施形態において、複数のCC-ALF関連構文要素は、第7の構文要素が、Cr色成分に関するCC-ALFが現在のピクチャに対して有効にされていることを示すものとして取得される場合、第8の構文要素を含み、第8の構文要素は、ピクチャヘッダから取得され、第8の構文要素は、現在のピクチャ内のすべてのスライスのCr色成分に関連付けられたパラメータセットを示す。第8の構文要素は、pic_cross_component_alf_cr_aps_idで示され得るが、これは、単なる一例である。この構文要素は、フィルタリング中に使用されるべき関連パラメータに関する情報を提供することができる。

【0060】

第4の構文要素、第5の構文要素、第6の構文要素、第7の構文要素、第8の構文要素、および第9の構文要素は、第3の構文要素が、CC-ALFがビットストリームに関連付けられたビデオシーケンスの現在のピクチャに対して有効にされていることを示すものとして取得される場合、取得されることがさらに規定され得る。

【0061】

一実施形態において、複数のCC-ALF関連構文要素は、第2の構文要素が、CC-ALFが有効にされていることを示すものとして取得される場合、第10の構文要素を含み、第10の構文要素は、スライスヘッダから取得され、第10の構文要素は、Cb色成分に関するCC-ALFがビットストリームに関連付けられたビデオシーケンスの現在のピクチャの現在のスライスに対して有効にされているかどうかを示す。この構文要素は、本開示を限定することを意図することなく、slice_cross_component_alf_cb_enabled_flagと呼ばれる場合がある。これによって、デコーダは、CC-ALFが提供され得るCb色成分に対して有効にされるべきであるかどうかを決定することができる。

【0062】

第10の構文要素が1の値を有する場合、それは、Cb色成分に関するCCALFが現在のスライスに対して有効にされていることを示し、および/または第10の構文要素が0の値を有する場合、それは、Cb色成分に関するCCALFが現在のスライスに対して無効にされていることを示すことがさらに規定され得る。

【0063】

一実施形態において、複数のCC-ALF関連構文要素は、第10の構文要素が、Cb色成分に関するCC-ALFが現在のスライスに対して有効にされていることを示すものとして取得される場合、第11の構文要素を含み、第10の構文要素は、スライスヘッダから取得され、第10の構文要素は、現在のスライスのCb色成分が参照するパラメータセットを指定する。この構文要素は、pic_cross_component_alf_cb_aps_idで示される場合があるが、これは、本開示を限定することを意図していない。

【0064】

複数のCC-ALF関連構文要素は、第2の構文要素が、CC-ALFが有効にされていることを示すものとして取得される場合、第12の構文要素を含み、第12の構文要素は、スライスヘッダから取得され、第12の構文要素は、Cb色成分に関するCCALFがビットストリームに関連付けられたビデオシーケンスの現在のピクチャの現在のスライスに対して有効にされているかどうかを示す。第12の構文要素は、たとえば、slice_cross_component_alf_cr_enabled_flagで示され得ることが規定され得る。

【0065】

より具体的な実施形態において、第12の構文要素が1の値を有する場合、それは、Cr色成分に関するCCALFが現在のスライスに対して有効にされていることを示し、および/または第12の構文要素が0の値を有する場合、それは、Cr色成分に関するCCALFが現在のスライスに対して無効にされていることを示す。

【0066】

複数のCC-ALF関連構文要素は、第12の構文要素が、Cr色成分に関するCC-ALFが現在のスライスに対して有効にされていることを示すものとして取得される場合、第13の構文

10

20

30

40

50

要素を含み、第13の構文要素は、スライスヘッダから取得され、第13の構文要素は、現在のスライスのCr色成分が参照するパラメータセットを指定することがさらに規定され得る。第13の構文要素は、たとえば、`slice_cross_component_alf_cr_aps_id`で示され得る。これは、Cr色成分に関連付けられたフィルタリングのために使用されるべきパラメータに関する情報を効率的に提供することができる。

【0067】

一実施形態において、第2の構文要素は、第1の構文要素が第1の値を有する場合に取得されるか、または第2の構文要素は、少なくとも第1の構文要素の値に基づいて取得される。

【0068】

複数のCC-ALF関連構文要素は、SPSレベルから取得される第14の構文要素を含み、第14の構文要素は、CC-ALFへの入力のタイプを示すことがさらに規定され得る。第14の構文要素は、`ChromaArrayType`で示され得るが、これは、本開示を限定していない。

【0069】

より具体的には、第2の構文要素は、第1の構文要素が第1の値を有し、かつ第14の構文要素が第2の値を有する場合に取得され得る。この第2の値は、CC-ALFが有効にされるべきことを示すか、またはそのように示すために使用され得る値とは異なる任意の値であり得る。

【0070】

より具体的には、第2の構文要素は、第1の構文要素が1に等しい値を有し、かつ第14の構文要素が0に等しくない値を有する場合に取得されることが規定され得る。

【0071】

上記の実施形態のいずれかについて、CC-ALFが、適応ループフィルタプロセスの一部として動作し、少なくとも1つのクロマ成分を精緻化するためにルマサンプル値を利用することがさらに規定され得る。

【0072】

本開示は、ビデオデータを符号化するためのデバイスにさらに関係し、デバイスは、ビデオデータメモリと、ビデオエンコーダと

を備え、ビデオエンコーダが、クロスコンポーネント適応ループフィルタCC-ALFを適用してクロマ成分を精緻化することと、複数のCC-ALF関連構文要素を含むビットストリームを生成することと、複数のCC-ALF関連構文要素がCCALF関連情報を示す、こととを実行するように構成され、複数のCC-ALF関連構文要素が、シーケンスパラメータセット(SPS)レベル、ピクチャヘッダ、またはスライスヘッダのうちの任意の1つまたは複数においてシグナリングされ、複数のCC-ALF関連構文要素が、第1の構文要素であって、クロスコンポーネント適応ループフィルタを含む適応ループフィルタ(ALF)がシーケンスレベルにおいて有効にされているかどうかを示し、SPSレベルにおいてシグナリングされる、第1の構文要素と、第2の構文要素であって、クロスコンポーネント適応ループフィルタがシーケンスレベルにおいて有効にされているかどうかを示し、SPSレベルにおいてシグナリングされる、第2の構文要素とを含む。これにより、上記で言及した符号化方法の利点は、たとえば、ビデオを符号化するためのエンコーダに提供される。

【0073】

本開示は、ビデオデータを復号するためのデバイスにも関係し、デバイスは、ビデオデータメモリと、

ビデオデコーダであって、ビデオデコーダが、ビットストリームからの複数のクロスコンポーネント適応ループフィルタCC-ALF関連構文要素を解析することと、複数の構文要素、複数のCC-ALF関連構文要素が、シーケンスパラメータセット(SPS)レベル、ピクチャヘッダ、またはスライスヘッダのうちの任意の1つまたは複数から取得され、複数のCC-ALF関連構文要素が、第1の構文要素であって、クロスコンポーネント適応ループフィルタを含む適応ループフィルタ(ALF)がシーケンスレベルにおいて有効にされているか

10

20

30

40

50

どうかを示し、SPSレベルにおいてシグナリングされる、第1の構文要素と、第2の構文要素であって、クロスコンポーネント適応ループフィルタがシーケンスレベルにおいて有効にされているかどうかを示し、SPSレベルにおいてシグナリングされる、第2の構文要素を含む、ことと、複数のCC-ALF関連構文要素のうちの少なくとも1つを使用してCC-ALFプロセスを実行することとを実行するように構成された、ビデオデコーダと

を備える。これにより、ビデオの信頼性の高い復号を取得しながら、ビットストリームの低減したサイズの利点の実現される。

【0074】

さらに、ビデオを符号化するためのエンコーダが提供され、エンコーダは、上記の実施形態のうちのいずれかによる方法を実行するための処理回路を備える。これにより、上記で言及した符号化方法の利点は、たとえば、ビデオを符号化するためのエンコーダに提供される。

【0075】

さらに、ビットストリームを復号するためのデコーダが提供され、デコーダは、上記の実施形態のうちのいずれかによる方法を実行するための処理回路を備える。

【0076】

これにより、ビデオの信頼性の高い復号を取得しながら、ビットストリームの低減したサイズの利点の実現される。

【0077】

本開示内でされる提供されるのは、コンピューティングデバイスによって実行されると、コンピューティングデバイスに、上記の実施形態のうちのいずれかによる方法を実行させるコンピュータ実行可能命令を記憶したコンピュータ可読記憶媒体である。

【0078】

本開示は、複数のCC-ALF関連構文要素を含むことによって、ビデオ信号のための符号化ビットストリームをさらに提供し、複数のCC-ALF関連構文要素は、CC-ALF関連情報を示し、

複数のCC-ALF関連構文要素は、シーケンスパラメータセット(SPS)レベル、ピクチャヘッダ、またはスライスヘッダのうちの任意の1つまたは複数においてシグナリングされ、

複数のCC-ALF関連構文要素は、第1の構文要素であって、クロスコンポーネント適応ループフィルタを含む適応ループフィルタ(ALF)がシーケンスレベルにおいて有効にされているかどうかを示し、SPSレベルにおいてシグナリングされる、第1の構文要素と、第2の構文要素であって、クロスコンポーネント適応ループフィルタがシーケンスレベルにおいて有効にされているかどうかを示し、SPSレベルにおいてシグナリングされる、第2の構文要素とを含む。

【0079】

CC-ALFを確実な方法で適用する際に復号において使用されるべき情報を提供しながら、ビットストリームは、サイズを低減され得る。

【0080】

1つまたは複数の実施形態の詳細は、添付図面および以下の説明において記載されている。他の特徴、目的、および利点は、説明、図面、および特許請求の範囲から明らかになるであろう。

【0081】

以下において、本発明の実施形態について、添付の図および図面を参照してより詳細に説明する。

【図面の簡単な説明】

【0082】

【図1A】本発明の実施形態を実装するように構成されたビデオコーディングシステムの一例を示すブロック図である。

【図1B】本発明の実施形態を実装するように構成されたビデオコーディングシステムの別の例を示すブロック図である。

10

20

30

40

50

【図2】本発明の実施形態を実装するように構成されたビデオエンコーダの一例を示すブロック図である。

【図3】本発明の実施形態を実装するように構成されたビデオコーダの例示的な構造を示すブロック図である。

【図4】符号化装置または復号装置の一例を示すブロック図である。

【図5】符号化装置または復号装置の別の例を示すブロック図である。

【図6】(a)は他のループフィルタに対するCC ALFの配置の図であり、(b)および(c)はダイヤモンド形フィルタの図である。

【図7】従来技術においてすべてのスライスヘッダがCCALFデータを送信しなければならないことを示す図である。

10

【図8】CC-ALFの改善された構文要素の一例を示す図である。

【図9A】ルマサンプルおよびクロマサンプルの公称の垂直および水平相対位置を示す概念図である。

【図9B】同じ場所に配置されたルマブロックおよびクロマブロックを示す概略図である。

【図10】コンテンツ配信サービスを実現するコンテンツ供給システム3100の例示的な構造を示すブロック図である。

【図11】端末デバイスの一例の構造を示すブロック図である。

【図12】エンコーダの一例を示すブロック図である。

【図13】デコーダの一例を示すブロック図である。

【図14】一実施形態による、ビデオを符号化する方法のフロー図である。

20

【図15】一実施形態による、ビデオを復号する方法のフロー図である。

【図16】データ構造5000、すなわちビデオビットストリーム500の概略図である。

【発明を実施するための形態】

【0083】

以下において、同一の参照符号は、他の方法で明示的に指定されない場合、同一のまたは少なくとも機能的に同等の特徴を指す。

【0084】

以下の定義は、参照のためのものである。

- ・ コーディングブロック:CTBのコーディングブロックへの分割がパーティション分割であるような、MおよびNのいくつかの値に関するサンプルのM×Nブロック。

30

- ・ コーディングツリーブロック(CTB(coding tree block)):コンポーネントのCTBへの分割がパーティション分割であるような、Nのいくつかの値に関するサンプルのN×Nブロック。

- ・ コーディングツリーユニット(CTU(coding tree unit)):ルマサンプルのCTB、3つのサンプル配列を有するピクチャのクロマサンプルの2つの対応するCTB、またはモノクロピクチャもしくはサンプルをコーディングするために使用される3つの別個の色平面および構文構造を使用してコーディングされたピクチャのサンプルのCTB。

- ・ コーディングユニット(CU(coding unit)):ルマサンプルのコーディングブロック、3つのサンプル配列を有するピクチャのクロマサンプルの2つの対応するコーディングブロック、またはモノクロピクチャもしくはサンプルをコーディングするために使用される3つの別個の色平面および構文構造を使用してコーディングされたピクチャのサンプルのコーディングブロック。

40

- ・ コンポーネント:4:2:0、4:2:2、もしくは4:4:4色フォーマットにおけるピクチャを構成する3つの配列(ルマおよび2つのクロマ)のうちの1つからの配列もしくは単一のサンプル、またはモノクロフォーマットにおけるピクチャを構成する配列または配列の単一のサンプル。

【0085】

以下の説明において、本開示の一部を形成し、例示として、本発明の実施形態の特定の態様、または本発明の実施形態が使用され得る特定の態様を示す添付の図への参照がなされる。本発明の実施形態は、他の態様において使用され得、図中に示されていない構造的

50

または論理的变化を含むことができることが理解される。したがって、以下の詳細な説明は、限定的な意味において解釈されるべきではなく、本発明の範囲は、添付の特許請求の範囲によって定義される。

【0086】

たとえば、説明されている方法に関連する開示は、方法を実行するように構成された対応するデバイスまたはシステムにも当てはまる場合があり、逆もまた同様であることが理解される。たとえば、1つまたは複数の特定の方法ステップが説明されている場合、対応するデバイスは、そのような1つまたは複数のユニットが明示的に説明されていないか、または図中に示されていない場合であっても、説明されている1つまたは複数の方法ステップを実行するための1つまたは複数のユニット、たとえば、機能ユニット(たとえば、1つもしくは複数のステップを実行する1つのユニット、または複数のステップのうちの1つもしくは複数の機能を各々が実行する複数のユニット)を含み得る。他方において、たとえば、特定の装置が1つまたは複数のユニット、たとえば、機能ユニットに基づいて説明されている場合、対応する方法は、そのような1つまたは複数のステップが明示的に説明されていないか、または図中に示されていない場合であっても、1つまたは複数のユニットの機能を実行するための1つのステップ(たとえば、1つもしくは複数のユニットの機能を実行する1つのステップ、または複数のユニットのうちの1つもしくは複数の機能を各々が実行する複数のステップ)を含み得る。さらに、本明細書に記載の様々な例示的な実施形態および/または態様の特徴は、特に他の方法で断らない限り、互いに組み合わせられ得ることが理解される。

【0087】

ビデオコーディングは、典型的には、ビデオまたはビデオシーケンスを形成するピクチャのシーケンスの処理を指す。「ピクチャ」という用語の代わりに、「フレーム」または「画像」という用語が、ビデオコーディングの分野において同義語として使用され得る。ビデオコーディング(または一般にコーディング)は、ビデオ符号化とビデオ復号の2つの部分を含む。ビデオ符号化は、ソース側において実行され、典型的には、(より効率的な記憶および/または伝送のために)ビデオピクチャを表現するために必要なデータの量を低減するために、(たとえば、圧縮によって)元のビデオピクチャを処理することを含む。ビデオ復号は、宛先側において実行され、典型的には、ビデオピクチャを再構築するために、エンコーダと比較して逆の処理を含む。ビデオピクチャ(または一般にピクチャ)の「コーディング」に言及する実施形態は、ビデオピクチャまたはそれぞれのビデオシーケンスの「符号化」または「復号」に関するものと理解されるべきである。符号化部分と復号部分の組合せは、コーデック(コーディングおよび復号)とも呼ばれる。

【0088】

ロスレスビデオコーディングの場合、元のビデオピクチャが再構築され得、すなわち、再構築されたビデオピクチャは、(記憶または伝送中に伝送損失または他のデータ損失がないと仮定して)元のビデオピクチャと同じ品質を有する。ロッキービデオコーディングの場合、デコーダにおいて完全に再構築することができないビデオピクチャを表現するデータの量を低減するために、たとえば、量子化によるさらなる圧縮が実行され、すなわち、再構築されたビデオピクチャの品質は、元のビデオピクチャの品質と比較して低いかまたは悪い。

【0089】

いくつかのビデオコーディング規格は、「ロッキーハイブリッドビデオコーデック」のグループに属する(すなわち、サンプルドメインにおける空間的および時間的予測と、変換ドメインにおける量子化を適用するための2D変換コーディングとを組み合わせる)。ビデオシーケンスの各ピクチャは、典型的には、重複しないブロックのセットにパーティション分割され、コーディングは、典型的には、ブロックレベルにおいて実行される。言い換えれば、エンコーダにおいて、ビデオは、典型的には、たとえば、予測ブロックを生成するために空間的(ピクチャ内)予測および/または時間的(ピクチャ間)予測を使用し、残差ブロックを取得するために、現在のブロック(現在処理されている/処理されるべきブロック)

から予測ブロックを減算し、残差ブロックを変換し、伝送されるべきデータの量を低減する(圧縮)ために、変換ドメインにおいて残差ブロックを量子化することによって、ブロック(ビデオブロック)レベルにおいて処理され、すなわち、符号化され、一方、デコーダ側において、表現のために現在のブロックを再構築するために、エンコーダと比較して逆処理が、符号化および圧縮されたブロックに適用される。さらに、エンコーダは、両方が後続のブロックを処理、すなわちコーディングするために同一の予測(たとえば、イントラ予測およびインター予測)および/または再構築を生成するように、デコーダ処理ループを複製する。

【0090】

以下において、ビデオコーディングシステム10、ビデオエンコーダ20、およびビデオデコーダ30の実施形態について、図1から図3に基づいて説明する。

10

【0091】

図1Aは、例示的なビデオコーディングシステム10、たとえば、この本出願の技法を利用し得るビデオコーディングシステム10(または略してコーディングシステム10)を示す概略ブロック図である。ビデオコーディングシステム10のビデオエンコーダ20(または略してエンコーダ20)およびビデオデコーダ30(または略してデコーダ30)は、本出願において説明されている様々な例による技法を実行するように構成され得るデバイスの例を表す。

【0092】

図1Aにおいて示すように、コーディングシステム10は、たとえば、符号化ピクチャデータ13を復号するために宛先デバイス14に符号化ピクチャデータ21を提供するように構成されたソースデバイス12を備える。

20

【0093】

ソースデバイス12は、エンコーダ20を備え得、ピクチャソース16と、プリプロセッサ(または前処理ユニット)18、たとえば、ピクチャプリプロセッサ18と、通信インターフェースまたは通信ユニット22とを、追加で、すなわちオプションで備え得る。

【0094】

ピクチャソース16は、任意の種類のピクチャキャプチャデバイス、たとえば、実世界のピクチャをキャプチャするためのカメラ、および/または任意の種類のピクチャ生成デバイス、たとえば、コンピュータアニメーションピクチャを生成するためのコンピュータグラフィックスプロセッサ、または実世界のピクチャ、コンピュータ生成ピクチャ(たとえば、スクリーンコンテンツ、仮想現実(VR(virtual reality))ピクチャ)、および/もしくはそれらの任意の組合せ(たとえば、拡張現実(AR(augmented reality))ピクチャ)を取得および/もしくは提供するための任意の種類の他のデバイスを備えるか、またはそれらであり得る。ピクチャソースは、前述のピクチャのうちのいずれかを記憶する任意の種類のメモリまたはストレージであり得る。

30

【0095】

プリプロセッサ18、および前処理ユニット18によって実行される処理と区別して、ピクチャまたはピクチャデータ17は、生のピクチャまたは生のピクチャデータ17と呼ばれる場合もある。

【0096】

40

プリプロセッサ18は、(生の)ピクチャデータ17を受信し、前処理されたピクチャ19または前処理されたピクチャデータ19を取得するために、ピクチャデータ17に対して前処理を実行するように構成され得る。プリプロセッサ18によって実行される前処理は、たとえば、トリミング、(たとえば、RGBからYCbCrへの)色フォーマットコンバージョン、色補正、またはノイズ除去を含み得る。前処理ユニット18は、オプションの構成要素であることが理解され得る。

【0097】

ビデオエンコーダ20は、前処理されたピクチャデータ19を受信し、符号化ピクチャデータ21を提供するように構成され得る(さらなる詳細について、たとえば、図2に基づいて以下で説明する)。

50

【0098】

ソースデバイス12の通信インターフェース22は、符号化ピクチャデータ21を受信し、符号化ピクチャデータ21(またはその任意のさらなる処理されたバージョン)を、通信チャネル13を介して、記憶または直接の再構築のために別のデバイス、たとえば、宛先デバイス14または任意の他のデバイスに送信するように構成され得る。

【0099】

宛先デバイス14は、デコーダ30(たとえば、ビデオデコーダ30)を備え得、通信インターフェースまたは通信ユニット28と、ポストプロセッサ32(または後処理ユニット32)と、ディスプレイデバイス34とを追加で、すなわちオプションで備え得る。

【0100】

宛先デバイス14の通信インターフェース28は、たとえば、ソースデバイス12から直接、または任意の他のソース、たとえば、記憶デバイス、たとえば、符号化ピクチャデータ記憶デバイスから、符号化ピクチャデータ21(またはその任意のさらなる処理されたバージョン)を受信し、符号化ピクチャデータ21をデコーダ30に提供するように構成され得る。

【0101】

通信インターフェース22および通信インターフェース28は、ソースデバイス12と宛先デバイス14との間の直接通信リンク、たとえば、直接の有線もしくはワイヤレス接続を介して、または任意の種類のネットワーク、たとえば、有線もしくはワイヤレスネットワークもしくはその任意の組合せ、もしくは任意の種類のプライベートおよびパブリックネットワーク、もしくはその任意の組合せを介して、符号化ピクチャデータ21または符号化データ13を送信または受信するように構成され得る。

【0102】

通信インターフェース22は、たとえば、符号化ピクチャデータ21を適切なフォーマット、たとえば、パケットにパッケージ化し、および/または通信リンクもしくは通信ネットワークを介する伝送のために、任意の種類の伝送符号化または処理を使用して、符号化ピクチャデータを処理するように構成され得る。

【0103】

通信インターフェース22の対応物を形成する通信インターフェース28は、たとえば、送信されたデータを受信し、符号化ピクチャデータ21を取得するために、任意の種類の対応する伝送復号もしくは処理および/またはパッケージ解除を使用して、送信データを処理するように構成され得る。

【0104】

通信インターフェース22と通信インターフェース28の両方または少なくとも一方は、ソースデバイス12から宛先デバイス14を指す図1A中の通信チャネル13に関する矢印によって示されているように単方向通信インターフェースとして、または双方向通信インターフェースとして構成され得、たとえば、メッセージを送信および受信し、たとえば、接続を設定し、通信リンクおよび/またはデータ伝送、たとえば、符号化ピクチャデータ伝送に関連する任意の他の情報を確認および交換するように構成され得る。

【0105】

デコーダ30は、符号化ピクチャデータ21を受信し、復号されたピクチャデータ31または復号されたピクチャ31を提供するように構成され得る(さらなる詳細について、たとえば、図3または図5に基づいて、以下で説明する)。

【0106】

宛先デバイス14のポストプロセッサ32は、後処理されたピクチャデータ33、たとえば、後処理されたピクチャ33を取得するために、復号されたピクチャデータ31(再構築されたピクチャデータとも呼ばれる)、たとえば、復号されたピクチャ31を後処理するように構成され得る。後処理ユニット32によって実行される後処理は、たとえば、ディスプレイデバイス34によって表示するための復号されたピクチャデータ31を準備するために、たとえば、(たとえば、RGBからYCbCrへの)色フォーマットコンバージョン、色補正、トリ

10

20

30

40

50

ミング、または再サンプリング、または任意の他の処理を含み得る。

【0107】

宛先デバイス14のディスプレイデバイス34は、たとえば、ユーザまたは視聴者に対してピクチャを表示するための後処理されたピクチャデータ33を受信するように構成され得る。ディスプレイデバイス34は、再構築されたピクチャを表現するための任意の種類のディスプレイ、たとえば、一体型または外部ディスプレイまたはモニタであるか、またはそれを備え得る。ディスプレイは、たとえば、液晶ディスプレイ(LCD(liquid crystal display))、有機発光ダイオード(OLED(organic light emitting diode))プラズマディスプレイ、プロジェクタ、マイクロLEDディスプレイ、液晶オンシリコン(LCoS(liquid crystal display))、デジタルライトプロセッサ(DLP(digital light processor))、または任意の種類の他のディスプレイを備え得る。

10

【0108】

図1Aは、ソースデバイス12および宛先デバイス14を別個のデバイスとして示しているが、デバイスの実施形態は、両方または両方の機能、ソースデバイス12または対応する機能および宛先デバイス14または対応する機能も含み得る。そのような実施形態において、ソースデバイス12または対応する機能および宛先デバイス14または対応する機能は、同じハードウェアおよび/もしくはソフトウェアを使用して、または別個のハードウェアおよび/もしくはソフトウェアもしくはそれらの任意の組合せによって実装され得る。

【0109】

ソースデバイスおよび/または宛先デバイスは、専用ハードウェアおよび/またはソフトウェアを使用してさらに実装され得る。たとえば、これらのデバイスのうちの一方または両方は、上記および下記で言及する機能のうちの1つまたは複数を実現するために特別に設計されたハードウェアを使用して実装され得る。代替的または追加的に、上記および下記で説明する機能のうちの1つまたは複数は、プロセッサなどの汎用ハードウェア上で実行され得る特別に設計されたソフトウェアを使用して実装され得る。さらに、上記の組合せも想定され、ソースデバイスおよび/または宛先デバイスは、1つまたは複数の機能を実現するための専用ハードウェアと、1つまたは複数の他の機能を実現するためのソフトウェアとの組合せを使用して実装され得る。

20

【0110】

説明に基づいて当業者には明らかであるように、図1Aにおいて示すようなソースデバイス12および/または宛先デバイス14内の異なるユニットの機能または機能の存在および(正確な)分割は、実際のデバイスおよび用途に応じて変化し得る。

30

【0111】

エンコーダ20(たとえば、ビデオエンコーダ20)またはデコーダ30(たとえば、ビデオデコーダ30)、またはエンコーダ20とデコーダ30の両方は、1つまたは複数のマイクロプロセッサ、デジタル信号プロセッサ(DSP(digital signal processor))、特定用途向け集積回路(ASIC(application-specific integrated circuit))、フィールドプログラマブルゲートアレイ(FPGA(field-programmable gate array))、ディスクリートロジック、ハードウェア、ビデオコーディング専用、またはそれらの任意の組合せなどの、図1Bにおいて示すような処理回路を介して実装され得る。エンコーダ20は、図2のエンコーダ20に関して論じたような様々なモジュールおよび/または本明細書に記載の任意の他のエンコーダシステムもしくはサブシステムを具体化するために、処理回路46を介して実装され得る。デコーダ30は、図3のデコーダ30に関して論じたような様々なモジュールおよび/または本明細書に記載の任意の他のデコーダシステムもしくはサブシステムを具体化するために、処理回路46を介して実装され得る。処理回路は、後に論じるように、様々な動作を実行するように構成され得る。図5において示すように、技法が部分的にソフトウェアにおいて実装される場合、デバイスは、ソフトウェアのための命令を、適切な非一時的なコンピュータ可読記憶媒体内に記憶し得、本開示の技法を実行するために1つまたは複数のプロセッサを使用するハードウェアにおいて命令を実行し得る。ビデオエンコーダ20およびビデオデコーダ30のいずれも、たとえば、図1Bにおいて示すように、単一のデバイス内の組

40

50

み合わされたエンコーダ/デコーダ(コーデック)の一部として統合され得る。

【0112】

ソースデバイス12および宛先デバイス14は、任意の種類のハンドヘルドまたは固定デバイス、たとえば、ノートブックまたはラップトップコンピュータ、携帯電話、スマートフォン、タブレットまたはタブレットコンピュータ、カメラ、デスクトップコンピュータ、セットトップボックス、テレビ、ディスプレイデバイス、デジタルメディアプレーヤ、ビデオゲームコンソール、ビデオストリーミングデバイス(コンテンツサービスサーバまたはコンテンツ配信サーバなど)、ブロードキャスト受信機デバイス、ブロードキャスト送信機デバイスなどを含む、多種多様なデバイスのうちのいずれかを含み得、オペレーティングシステムを使用しないか、または任意の種類のオペレーティングシステムを使用し得る。場合によっては、ソースデバイス12および宛先デバイス14は、ワイヤレス通信のために装備され得る。したがって、ソースデバイス12および宛先デバイス14は、ワイヤレス通信デバイスであり得る。

10

【0113】

場合によっては、図1Aにおいて示すビデオコーディングシステム10は、単なる例であり、本出願の技法は、符号化デバイスと復号デバイスとの間のデータ通信を必ずしも含まないビデオコーディング設定(たとえば、ビデオ符号化またはビデオ復号)に適用され得る。他の例において、データは、ローカルメモリから取得される、ネットワークを介してストリーミングされる、などをされる。ビデオ符号化デバイスは、データを符号化し、メモリに記憶し得、および/またはビデオ復号デバイスは、メモリからデータを取り出し、復号し得る。いくつかの例において、符号化および復号は、互いに通信しないが、単にデータをメモリに符号化するおよび/またはメモリからデータを取り出し、復号するデバイスによって実行される。

20

【0114】

説明の便宜のために、本発明の実施形態について、たとえば、高効率ビデオコーディング(HEVC)、またはITU-Tビデオコーディングエキスパートグループ(VECEG(Video Coding Experts Group))およびISO/IECモーションピクチャエキスパートグループ(MPEG(Motion Picture Experts Group))のジョイントコラボレーションチームオンビデオコーディング(JCT-VC(Joint Collaboration Team on Video Coding))によって開発された次世代ビデオコーディング規格、多用途ビデオコーディング(VVC)の参照ソフトウェアを参照することによって本明細書で説明する。当業者は、本発明の実施形態がHEVCまたはVVCに限定されないことを理解するであろう。

30

【0115】

エンコーダおよび符号化方法

図2は、本開示の技法を実装するように構成され得る例示的なビデオエンコーダ20の概略ブロック図を示す。図2の例において、ビデオエンコーダ20は、入力201(または入力インターフェース201)と、残差計算ユニット204と、変換処理ユニット206と、量子化ユニット208と、逆量子化ユニット210と、逆変換処理ユニット212と、再構築ユニット214と、ループフィルタユニット220と、復号ピクチャバッファ(DPB(decoded picture buffer))230と、モード選択ユニット260と、エントロピー符号化ユニット270と、出力272(または出力インターフェース272)とを備える。モード選択ユニット260は、インター予測ユニット244と、イントラ予測ユニット254と、パーティション分割ユニット262とを含み得る。インター予測ユニット244は、動き推定ユニットと、動き補償ユニット(図示せず)とを含み得る。図2において示すビデオエンコーダ20は、ハイブリッドビデオエンコーダ、またはハイブリッドビデオコーデックによるビデオエンコーダと呼ばれる場合もある。

40

【0116】

残差計算ユニット204、変換処理ユニット206、量子化ユニット208、モード選択ユニット260は、エンコーダ20の順方向信号経路を形成するものと呼ばれる場合があり、一方、逆量子化ユニット210、逆変換処理ユニット212、再構築ユニット214、バッファ216、ループフィルタ220、復号ピクチャバッファ(DPB)230、インター予測ユニット244、

50

およびイントラ予測ユニット254は、ビデオエンコーダ20の逆方向信号経路を形成するものと呼ばれる場合があり、ビデオエンコーダ20の逆方向信号経路は、デコーダの信号経路に対応する(図3におけるビデオデコーダ30を参照)。逆量子化ユニット210、逆変換処理ユニット212、再構築ユニット214、ループフィルタ220、復号ピクチャバッファ(DPB)230、インター予測ユニット244、およびイントラ予測ユニット254は、ビデオエンコーダ20の「ビルトインデコーダ」を形成するものとも呼ばれる。

【0117】

ピクチャおよびピクチャのパーティション分割(ピクチャおよびブロック)

エンコーダ20は、たとえば、入力201を介して、ピクチャ17(またはピクチャデータ17)、たとえば、ビデオまたはビデオシーケンスを形成するピクチャのシーケンスのピクチャを受信するように構成され得る。受信されたピクチャまたはピクチャデータは、前処理されたピクチャ19(または前処理されたピクチャデータ19)でもあり得る。簡単にするために、以下の説明は、ピクチャ17を参照する。ピクチャ17は、(特に、現在のピクチャを、他のピクチャ、たとえば、同じビデオシーケンス、すなわち、現在のピクチャも含むビデオシーケンスの以前に符号化および/または復号されたピクチャと区別するために、ビデオコーディングにおいて)現在のピクチャまたはコーディングされるべきピクチャとも呼ばれる場合がある。

【0118】

(デジタル)ピクチャは、強度値を有するサンプルの2次元配列または行列であるか、またはそう見なされ得る。配列内のサンプルは、ピクセル(ピクチャ要素の短縮形)またはペルと呼ばれる場合もある。配列またはピクチャの水平および垂直方向(または軸)におけるサンプル数は、ピクチャのサイズおよび/または解像度を定義する。色の表現のため、典型的には、3つの色成分が用いられ、すなわち、ピクチャは、3つのサンプル配列で表現されるか、または3つのサンプル配列を含み得る。RGBフォーマットまたは色空間において、ピクチャは、対応する赤色、緑色、および青色のサンプル配列を備える。しかしながら、ビデオコーディングにおいて、各ピクセルは、典型的には、ルミナンスおよびクロミナンスフォーマットまたは色空間、たとえば、Y(ときには、代わりにLも使用される)によって示されるルミナンス成分と、CbおよびCrによって示される2つのクロミナンス成分を含むYCbCrにおいて表される。ルミナンス(略してルマ)成分Yは、(たとえば、グレースケールピクチャにおけるように)明るさまたはグレーレベル強度を表し、2つのクロミナンス(略してクロマ)成分CbおよびCrは、色度または色情報成分を表す。したがって、YCbCrフォーマットにおけるピクチャは、ルミナンスサンプル値(Y)のルミナンスサンプル配列と、クロミナンス値(CbおよびCr)の2つのクロミナンスサンプル配列とを備える。RGBフォーマットにおけるピクチャは、YCbCrフォーマットにコンバートまたは変換され得、その逆も同様であり、プロセスは、色変換またはコンバージョンとしても知られる。ピクチャがモノクロである場合、ピクチャは、ルミナンスサンプル配列のみを備え得る。したがって、ピクチャは、たとえば、モノクロフォーマットにおけるルマサンプルの配列、または4:2:0、4:2:2、および4:4:4色フォーマットにおけるルマサンプルの配列および2つの対応するクロマサンプルの配列であり得る。

【0119】

ビデオエンコーダ20の実施形態は、ピクチャ17を複数の(典型的には重複しない)ピクチャブロック203にパーティション分割するように構成されたピクチャパーティション分割ユニット(図2には示さず)を備え得る。これらのブロックは、ルートブロック、マクロブロック(H.264/AVC)、またはコーディングツリーブロック(CTB)もしくはコーディングツリーユニット(CTU)(H.265/HEVCおよびVVC)と呼ばれる場合もある。ピクチャパーティション分割ユニットは、ビデオシーケンスのすべてのピクチャについて同じブロックサイズと、ブロックサイズを定義する対応するグリッドとを使用するように構成されるか、またはピクチャ間、ピクチャのサブセット間、もしくはピクチャのグループ間でブロックサイズを変更し、各ピクチャを対応するブロックにパーティション分割するように構成され得る。

【0120】

さらなる実施形態において、ビデオエンコーダは、ピクチャ17のブロック203、たとえば、ピクチャ17を形成する1つの、いくつかの、またはすべてのブロックを直接受信するように構成され得る。ピクチャブロック203は、現在のピクチャブロック、またはコーディングされるべきピクチャブロックと呼ばれる場合もある。

【0121】

ピクチャ17と同様に、ピクチャブロック203は、再び、ピクチャ17よりも小さい寸法であるが、強度値(サンプル値)を有するサンプルの2次元配列または行列であるか、またはそのように見なされ得る。言い換えれば、ブロック203は、たとえば、1つのサンプル配列(たとえば、モノクロピクチャ17の場合、ルマ配列、またはカラーピクチャの場合、ルマもしくはクロマ配列)、または3つのサンプル配列(たとえば、カラーピクチャ17の場合、ルマ配列および2つのクロマ配列)、または適用される色フォーマットに応じて任意の他の数および/もしくは種類の配列を備え得る。ブロック203の水平および垂直方向(または軸)におけるサンプル数は、ブロック203のサイズを定義する。したがって、ブロックは、たとえば、サンプルの $M \times N$ (M 列 $\times$ N 行)配列、または変換係数の $M \times N$ 配列であり得る。

【0122】

図2において示すようなビデオエンコーダ20の実施形態は、ピクチャ17をブロックごとに符号化するように構成され得、たとえば、符号化および予測は、ブロック203ごとに実行される。

【0123】

図2において示すようなビデオエンコーダ20の実施形態は、スライス(ビデオスライスとも呼ばれる)を使用することによってピクチャをパーティション分割および/または符号化するようにさらに構成され得、ピクチャは、1つまたは複数のスライス(典型的には重複しない)を使用してパーティション分割または符号化され得、各スライスは、1つもしくは複数のブロック(たとえば、CTU)、またはブロックの1つもしくは複数のグループ(たとえば、タイル(H.265/HEVCおよびVVC)またはブリック(VVC))を備え得る。

【0124】

図2において示すようなビデオエンコーダ20の実施形態は、スライス/タイルグループ(ビデオタイルグループとも呼ばれる)および/またはタイル(ビデオタイルとも呼ばれる)を使用することによって、ピクチャをパーティション分割および/または符号化するようにさらに構成され得、ピクチャは、1つまたは複数のスライス/タイルグループ(典型的には重複しない)を使用してパーティション分割または符号化され得、各スライス/タイルグループは、たとえば、1つもしくは複数のブロック(たとえば、CTU)または1つもしくは複数のタイルを備え得、各タイルは、たとえば、矩形形状であり得、1つまたは複数のブロック(たとえば、CTU)、たとえば、完全なまたは断片のブロックを備え得る。

【0125】

残差計算

残差計算ユニット204は、サンプルドメインにおける残差ブロック205を取得するために、たとえば、サンプルごと(ピクセルごと)にピクチャブロック203のサンプル値から予測ブロック265のサンプル値を減算することによって、ピクチャブロック203と予測ブロック265(予測ブロック265に関するさらなる詳細については、後に提供する)とに基づいて、残差ブロック205(残差205とも呼ばれる)を計算するように構成され得る。

【0126】

変換

変換処理ユニット206は、変換ドメインにおいて変換係数207を取得するために、残差ブロック205のサンプル値に対して変換、たとえば、離散コサイン変換(DCT(discrete cosine transform))または離散サイン変換(DST(discrete sine transform))を適用するように構成され得る。変換係数207は、変換残差係数と呼ばれる場合もあり、変換ドメインにおいて残差ブロック205を表し得る。

【0127】

変換処理ユニット206は、H.265/HEVCに対して指定された変換など、DCT/DSTの整数近似を適用するように構成され得る。直交DST変換と比較して、そのような整数近似は、典型的には、特定の係数によってスケーリングされる。順変換および逆変換によって処理される残差ブロックのノルムを維持するために、変換処理の一部として追加のスケーリング係数が適用される。スケーリング係数は、典型的には、シフト演算のための2のべき乗であるスケーリング係数、変換係数のビット深度、精度と実装コストとの間のトレードオフなどのような特定の制約に基づいて選択される。特定のスケーリング係数は、たとえば、逆変換処理ユニット212による逆変換(および、たとえば、ビデオデコーダ30における逆変換処理ユニット312による対応する逆変換)に対して指定され、たとえば、エンコーダ20における変換処理ユニット206による順変換のための対応するスケーリング係数が、それに応じて指定され得る。

10

【0128】

ビデオエンコーダ20(それぞれ変換処理ユニット206)の実施形態は、たとえば、ビデオデコーダ30が復号するために変換パラメータを受信および使用し得るように、たとえば、直接、またはエントロピー符号化ユニット270を介して符号化または圧縮された変換パラメータ、たとえば、1つまたは複数の変換のタイプを出力するように構成され得る。

【0129】

量子化

量子化ユニット208は、たとえば、スカラー量子化またはベクトル量子化を適用することによって、量子化された係数209を取得するために変換係数207を量子化するように構成され得る。量子化された係数209は、量子化された変換係数209または量子化された残差係数209と呼ばれる場合もある。

20

【0130】

量子化プロセスは、変換係数207の一部またはすべてに関連するビット深度を低減し得る。たとえば、nビット変換係数は、量子化中にmビット変換係数に切り捨てられ得、ここで、nは、mよりも大きい。量子化の程度は、量子化パラメータ(QP(quantization parameter))を調整することによって変更され得る。たとえば、スカラー量子化の場合、より細かいまたはより粗い量子化を達成するために、異なるスケーリングが適用され得る。より小さい量子化ステップサイズは、より細かい量子化に対応し、より大きい量子化ステップサイズは、より粗い量子化に対応する。適用可能な量子化ステップサイズは、量子化パラメータ(QP)によって示され得る。量子化パラメータは、たとえば、適用可能な量子化ステップサイズの事前定義されたセットへのインデックスであり得る。たとえば、小さい量子化パラメータは、細かい量子化(小さい量子化ステップサイズ)に対応し得、大きい量子化パラメータは、粗い量子化(大きい量子化ステップサイズ)に対応し得、またはその逆であり得る。量子化は、量子化ステップサイズによる除算を含み得、たとえば、逆量子化ユニット210による対応するおよび/または逆の脱量子化は、量子化ステップサイズによる乗算を含み得る。いくつかの規格、たとえば、HEVCによる実施形態は、量子化ステップサイズを決定するために量子化パラメータを使用するように構成され得る。一般に、量子化ステップサイズは、除算を含む方程式の固定小数点近似を使用して、量子化パラメータに基づいて計算され得る。量子化ステップサイズおよび量子化パラメータに関する方程式の固定小数点近似において使用されるスケーリングのために変更される可能性がある残差ブロックのノルムを復元するために、量子化および脱量子化のために追加のスケーリング係数が導入され得る。1つの例示的な実装形態において、逆変換および脱量子化のスケーリングは、組み合わされ得る。代替的には、カスタマイズされた量子化テーブルが使用され、たとえば、ビットストリームにおいてエンコーダからデコーダにシグナリングされ得る。量子化は、損失のある演算であり、損失は、量子化ステップサイズの増加に伴って増加する。

30

40

【0131】

ビデオエンコーダ20(それぞれ量子化ユニット208)の実施形態は、たとえばビデオデコーダ30が復号するために量子化パラメータを受信および適用し得るように、たとえば、直

50

接、またはエントロピー符号化ユニット270を介して符号化された量子化パラメータ(QP)を出力するように構成され得る。

【0132】

逆量子化

逆量子化ユニット210は、たとえば、量子化ユニット208と同じ量子化ステップサイズに基づいて、またはそれを使用して、量子化ユニット208によって適用された量子化方式の逆を適用することによって、脱量子化された係数211を取得するために、量子化された係数に対して量子化ユニット208の逆量子化を適用するように構成され得る。脱量子化された係数211は、脱量子化された残差係数211と呼ばれる場合もあり、典型的には、量子化による損失のために変換係数と同一ではないが、変換係数207に対応する。

10

【0133】

逆変換

逆変換処理ユニット212は、サンプルドメインにおいて再構築された残差ブロック213(または対応する脱量子化された係数213)を取得するために、変換処理ユニット206によって適用された変換の逆変換、たとえば、逆離散コサイン変換(DCT)、または逆離散サイン変換(DST)、または他の逆変換を適用するように構成され得る。再構築された残差ブロック213は、変換ブロック213と呼ばれる場合もある。

【0134】

再構築

再構築ユニット214(たとえば、加算器またはアナログ加算器214)は、たとえば、再構築された残差ブロック213のサンプル値と予測ブロック265のサンプル値とをサンプルごとに加算することによって、サンプルドメインにおいて再構築されたブロック215を取得するために、変換ブロック213(すなわち、再構築された残差ブロック213)を予測ブロック265に追加するように構成され得る。

20

【0135】

フィルタリング

ループフィルタユニット220(または略して「ループフィルタ」220)は、フィルタリングされたブロック221を取得するために、再構築されたブロック215をフィルタリングするように、または一般的に、フィルタリングされたサンプル値を取得するために、再構築されたサンプルをフィルタリングするように構成され得る。ループフィルタユニットは、たとえば、ピクセル遷移を滑らかにするか、または他の方法でビデオ品質を改善するように構成される。ループフィルタユニット220は、デブロッキングフィルタ、サンプル適応オフセット(SAO(sample-adaptive offset))フィルタ、または1つもしくは複数の他のフィルタ、たとえば、適応ループフィルタ(ALF)、ノイズ抑制フィルタ(NSF(noise suppression filter))、またはそれらの任意の組合せなどの、1つまたは複数のループフィルタを備え得る。一例において、ループフィルタユニット220は、デブロッキングフィルタ、SAOフィルタ、およびALFフィルタを備え得る。フィルタリングプロセスの順序は、デブロッキングフィルタ、SAO、およびALFであり得る。別の例において、クロマスケーリングによるルママッピング(LMCS(luma mapping with chroma scaling))(すなわち、適応ループ内リシェーパ)と呼ばれるプロセスが追加される。このプロセスは、デブロッキングの前に実行される。別の例において、デブロッキングフィルタプロセスは、内部サブブロックエッジ、たとえば、アフィンサブブロックエッジ、ATMVPサブブロックエッジ、サブブロック変換(SBT(sub-block transform))エッジ、およびイントラサブパーティション(ISP(intra sub-partition))エッジにも適用され得る。

30

40

【0136】

大きい「ブロック」に対して発生するブロッキングアーティファクトを効果的に除去するために、VVCは、より長いタップデブロッキングフィルタを使用する。ここで、「ブロック」という用語は、非常に一般的な方法で使用され、「変換ブロック(TB(transform block))、予測ブロック(PB(prediction block))、またはコーディングユニットブロック(CU(coding unit block))」を指す場合がある。より長いタップフィルタは、ルマ成分とク

50

ロマ成分の両方に適用される。ルマ成分のためのより長いタップフィルタは、エッジに対して垂直かつ隣接するサンプルの各ラインに対して最大7個のサンプルを変更し、デブロッキングの方向において32サンプル以上のサイズを有するブロックに対して適用され、すなわち、垂直エッジの場合、ブロック幅は、32サンプル以上であるべきであり、水平エッジの場合、ブロック高さは、32サンプル以上であるべきである。

【0137】

クロマのより長いタップフィルタは、所与のエッジに隣接する両方のブロックが8サンプル以上のサイズを有する場合にクロマブロックに適用され、エッジの両側において最大3個のサンプルを変更する。したがって、垂直エッジの場合、エッジに隣接する両方のブロックのブロック幅は、8サンプル以上であるべきであり、水平エッジの場合、エッジに隣接する両方のブロックのブロック高さは、8サンプル以上であるべきである。ループフィルタユニット220は、ループ内フィルタであるように図2において示されているが、他の構成において、ループフィルタユニット220は、ポストループフィルタとして実装され得る。フィルタリングされたブロック221は、フィルタリングされた再構築されたブロック221と呼ばれる場合もある。

【0138】

ビデオエンコーダ20(それぞれループフィルタユニット220)の実施形態は、たとえば、デコーダ30が、復号するために同じループフィルタパラメータまたはそれぞれのループフィルタを受信および適用し得るように、たとえば、直接、またはエントロピー符号化ユニット270を介して符号化されたループフィルタパラメータ(SAOフィルタパラメータまたはALFフィルタパラメータまたはLMCSパラメータなど)を出力するように構成され得る。

【0139】

復号ピクチャバッファ

復号ピクチャバッファ(DPB)230は、ビデオエンコーダ20によってビデオデータを符号化するために、参照ピクチャ、または一般に参照ピクチャデータを記憶するメモリであり得る。DPB230は、シンクロナス・ダイナミック・ランダム・アクセス・メモリ(SDRAM(synchronous dynamic random access memory))を含むダイナミックランダムアクセスメモリ(DRAM(dynamic random access memory))、磁気抵抗RAM(MRAM(magnetoresistive RAM))、抵抗RAM(RRAM(resistive RAM))、または他のタイプのメモリデバイスなどの、様々なメモリデバイスのうちのいずれかによって形成され得る。復号ピクチャバッファ(DPB)230は、1つまたは複数のフィルタリングされたブロック221を記憶するように構成され得る。復号ピクチャバッファ230は、同じ現在のピクチャまたは異なるピクチャ、たとえば、以前に再構築されたピクチャの他の以前にフィルタリングされたブロック、たとえば、以前に再構築およびフィルタリングされたブロック221を記憶するようにさらに構成され得、たとえば、インター予測のために、完全な以前に再構築された、すなわち復号されたピクチャ(ならびに対応する参照ブロックおよびサンプル)、および/または部分的に再構築された現在のピクチャ(ならびに対応する参照ブロックおよびサンプル)を提供し得る。復号ピクチャバッファ(DPB)230は、たとえば、再構築されたブロック215がループフィルタユニット220によってフィルタリングされていない場合、1つまたは複数のフィルタリングされていない再構築されたブロック215、もしくは一般にフィルタリングされていない再構築されたサンプル、または再構築されたブロックもしくはサンプルの任意の他のさらに処理されたバージョンを記憶するようにも構成され得る。

【0140】

モード選択(パーティション分割および予測)

モード選択ユニット260は、パーティション分割ユニット262と、インター予測ユニット244と、イントラ予測ユニット254とを備え、元のピクチャデータ、たとえば、元のブロック203(現在のピクチャ17の現在のブロック203)と、同じ(現在の)ピクチャのおよび/または1つもしくは複数の以前に復号されたピクチャからの、たとえば、復号ピクチャバッファ230もしくは他のバッファ(たとえば、ラインバッファ、図示せず)からの再構築されたピクチャ、たとえば、フィルタリングされたおよび/またはフィルタリングされていない

い再構築されたサンプルまたはブロックとを受信または取得するように構成される。再構築されたピクチャデータは、予測ブロック265または予測子265を取得するために、予測、たとえば、インター予測またはイントラ予測のための参照ピクチャブロックとして使用される。

【0141】

モード選択ユニット260は、現在のブロック予測モードに対するパーティション分割(パーティション分割なしを含む)と、予測モード(イントラまたはインター予測モード)とを決定または選択し、残差ブロック205の計算および再構築されたブロック215の再構築のために使用される対応する予測ブロック265を生成するように構成され得る。

【0142】

モード選択ユニット260の実施形態は、最良の一致、もしくは、言い換えれば、最小残差(最小残差は、伝送または記憶のためのよりよい圧縮を意味する)、もしくは最小シグナリングオーバーヘッド(最小シグナリングオーバーヘッドは、伝送または記憶のためのよりよい圧縮を意味する)を提供するか、または両方を考慮またはバランスさせるパーティション分割と予測モードとを(たとえば、モード選択ユニット260によってサポートされているもの、またはモード選択ユニット260に利用可能なものから)選択するように構成され得る。モード選択ユニット260は、レート歪み最適化(RDO(rate distortion optimization))に基づいて、パーティション分割と予測モードとを決定するように構成され得、すなわち、最小レート歪みを提供する予測モードを選択するように構成され得る。この文脈における「最良」、「最小」、「最適」などの用語は、必ずしも全体的な「最良」、「最小」、「最適」などを指すとは限らず、または「次善の選択」に潜在的につながるしきい値または他の制約を超える、または下回るが、複雑さと処理時間とを低減する値などの、終了または選択基準の達成を指す場合もある。

【0143】

言い換えれば、パーティション分割ユニット262は、ビデオシーケンスからのピクチャをコーディングツリーユニット(CTU)のシーケンスにパーティション分割するように構成され得、CTU203は、たとえば、四分木パーティション分割(QT(quad-tree-partitioning))、バイナリパーティション分割(BT(binary partitioning))、トリプルツリーパーティション分割(TT(triple-tree-partitioning))、またはそれらの任意の組合せを繰り返し使用し、たとえば、ブロックパーティション分割またはサブブロックごとに予測を実行するために、より小さいブロックパーティションまたはサブブロック(再びブロックを形成する)にさらにパーティション分割され得、モード選択は、パーティション分割されたブロック203のツリー構造の選択を含み、予測モードは、ブロックパーティションまたはサブブロックの各々に適用される。

【0144】

以下において、例示的なビデオエンコーダ20によって実行される(たとえば、パーティション分割ユニット260による)パーティション分割および(インター予測ユニット244およびイントラ予測ユニット254による)予測処理について、より詳細に説明する。

【0145】

パーティション分割

パーティション分割ユニット262は、ビデオシーケンスからのピクチャをコーディングツリーユニット(CTU)のシーケンスにパーティション分割するように構成され得、パーティション分割ユニット262は、コーディングツリーユニット(CTU)203をより小さいパーティション、たとえば、正方形または矩形のサイズのより小さいブロックにパーティション分割(または分割)し得る。3つのサンプル配列を有するピクチャの場合、CTUは、ルマサンプルの $N \times N$ ブロックと、クロマサンプルの2つの対応するブロックから構成される。CTUにおけるルマブロックの最大許容サイズは、開発中の多用途ビデオコーディング(VVC)においては 128×128 であるように指定されているが、将来においては 128×128 ではない値、たとえば、 256×256 であるように指定される可能性がある。ピクチャのCTUは、スライス/タイルグループ、タイル、またはブリックとしてクラスター化/グループ化され

得る。タイルは、ピクチャの矩形領域をカバーし、タイルは、1つまたは複数のブリックに分割され得る。ブリックは、タイル内のいくつかのCTU行から構成される。複数のブリックにパーティション分割されないタイルは、ブリックと呼ばれる場合がある。しかしながら、ブリックは、タイルの真のサブセットであり、タイルとは呼ばれない。VVCにおいてサポートされているタイルグループには2つのモード、すなわち、ラストスキャンスライス/タイルグループモードおよび矩形スライスモードが存在する。ラストスキャンタイルグループモードにおいて、スライス/タイルグループは、ピクチャのラストスキャンにおけるタイルのシーケンスを含む。矩形スライスモードにおいて、スライスは、ピクチャの矩形領域を集合的に形成するピクチャのいくつかのブリックを含む。矩形スライス内のブリックは、スライスのブリックラストスキャンの順序である。これらのより小さいブロック(サブブロックと呼ばれる場合もある)は、さらにより小さいパーティションにさらにパーティション分割され得る。これは、ツリーパーティション分割または階層ツリーパーティション分割とも呼ばれ、たとえば、ルートツリーレベル0(階層レベル0、深度0)におけるルートブロックは、再帰的にパーティション分割され得、たとえば、次のより低いツリーレベルの2つ以上のブロック、たとえば、ツリーレベル1(階層レベル1、深度1)におけるノードにパーティション分割され、たとえば、終了基準が満たされているので、たとえば、最大ツリー深度または最小ブロックサイズに達しているのでパーティション分割が終了するまで、これらのブロックは、次のより低いレベル、たとえば、ツリーレベル2(階層レベル2、深度2)の2つ以上のブロックに再びパーティション分割され得る。さらにパーティション分割されないブロックは、ツリーのリーフブロックまたはリーフノードとも呼ばれる。2つのパーティションへのパーティション分割を使用するツリーは、二分木(BT(binary-tree))と呼ばれ、3つのパーティションへのパーティション分割を使用するツリーは、三分木(TT(ternary-tree))と呼ばれ、4つのパーティションへのパーティション分割を使用するツリーは、四分木(QT(quad-tree))と呼ばれる。

【0146】

たとえば、コーディングツリーユニット(CTU)は、3つのサンプル配列を有するピクチャのルマサンプルのCTBと、クロマサンプルの2つの対応するCTBか、またはモノクロピクチャ、もしくはサンプルをコーディングするために使用される3つの別個の色平面と構文構造とを使用してコーディングされたピクチャのサンプルのCTBであるか、またはそれらを含み得る。これに対応して、コーディングツリーブロック(CTB)は、CTBへの構成要素の分割がパーティション分割であるように、Nのある値に対するサンプルの $N \times N$ ブロックであり得る。コーディングユニット(CU)は、3つのサンプル配列を有するピクチャのルマサンプルのコーディングブロックと、クロマサンプルの2つの対応するコーディングブロックか、またはモノクロピクチャ、もしくはサンプルをコーディングするために使用される3つの別個の色平面と構文構造とを使用してコーディングされたピクチャのサンプルのコーディングブロックであるか、またはそれらを含み得る。これに対応して、コーディングブロック(CB(coding block))は、CTBのコーディングブロックへの分割がパーティション分割であるように、MおよびNのある値に対するサンプルの $M \times N$ ブロックであり得る。

【0147】

実施形態において、たとえば、HEVCによれば、コーディングツリーユニット(CTU)は、コーディングツリーとして示される四分木構造を使用することによって、CUに分割され得る。ピクチャ間(時間的)またはピクチャ内(空間的)予測のどちらを使用してピクチャ領域をコーディングするかは決定は、リーフCUレベルにおいて行われる。各リーフCUは、PU分割タイプに従って1つ、2つ、または4つのPUにさらに分割され得る。1つのPU内で、同じ予測プロセスが適用され、関連情報は、PU単位でデコーダに送信される。PU分割タイプに基づく予測プロセスを適用することによって残差ブロックを取得した後、リーフCUは、CUに関するコーディングツリーと同様の別の四分木構造に従って、変換ユニット(TU(transform unit))にパーティション分割され得る。

【0148】

実施形態において、たとえば、多用途ビデオコーディング(VCC)と呼ばれる現在開発中

の最新のビデオコーディング規格によれば、バイナリ(binary)およびターナリー(ternary)を使用する組み合わされた四分木ネスト型マルチタイプツリーは、たとえば、コーディングツリーユニットをパーティション分割するために使用されるセグメンテーション構造を分割する。コーディングツリーユニット内のコーディングツリー構造において、CUは、正方形形状または矩形形状のいずれかを有することができる。たとえば、コーディングツリーユニット(CTU)は、最初に四分木によってパーティション分割される。次いで、四分木リーフノードは、マルチタイプツリー構造によってさらにパーティション分割され得る。マルチタイプツリー構造において4つの分割タイプ、垂直バイナリ分割(SPLIT_BT_VER)、水平バイナリ分割(SPLIT_BT_HOR)、垂直ターナリー分割(SPLIT_TT_VER)、および水平ターナリー分割(SPLIT_TT_HOR)が存在する。マルチタイプツリーリーフノードは、コーディングユニット(CU)と呼ばれ、CUが最大変換長に対して大き過ぎない限り、このセグメンテーションは、いかなるさらなるパーティション分割もなしに、予測および変換処理に使用される。これは、ほとんどの場合において、CU、PU、およびTUは、ネスト型マルチタイプツリーコーディングブロック構造を有する四分木において同じブロックサイズを有することを意味する。例外は、サポートされている最大変換長がCUの色成分の幅または高さよりも小さい場合に発生する。VCCは、ネスト型マルチタイプツリーコーディングツリー構造を有する四分木におけるパーティション分割情報の独自のシグナリングメカニズムを開発している。シグナリングメカニズムにおいて、コーディングツリーユニット(CTU)が、四分木のルートとして扱われ、最初に四分木構造によってパーティション分割される。次いで、各四分木リーフノード(それを可能にするのに十分に大きい場合)は、マルチタイプツリー構造によってさらにパーティション分割される。マルチタイプツリー構造において、ノードがさらにパーティション分割されるかどうかを示すために、第1のフラグ(mtt_split_cu_flag)がシグナリングされ、ノードがさらにパーティション分割される場合、分割方向を示すために、第2のフラグ(mtt_split_cu_vertical_flag)がシグナリングされ、次いで、分割がバイナリ分割であるかターナリー分割であるかを示すために、第3のフラグ(mtt_split_cu_binary_flag)がシグナリングされる。mtt_split_cu_vertical_flagおよびmtt_split_cu_binary_flagの値に基づいて、CUのマルチタイプツリースリッティングモード(MttSplitMode)は、事前定義されたルールまたはテーブルに基づいて、デコーダによって導出され得る。特定の設計、たとえば、VVCハードウェアデコーダにおける64×64ルマブロックおよび32×32クロマパイプライン設計の場合、図6に示すように、ルマコーディングブロックの幅または高さのいずれかが64よりも大きい場合、TT分割が禁止されることが留意されるべきである。クロマコーディングブロックの幅または高さのいずれかが32よりも大きい場合にも、TT分割は、禁止される。パイプライン設計は、ピクチャを、パイプライン内の非重複ユニットとして定義される仮想パイプラインデータユニット(VPDU(Virtual pipeline data unit))に分割する。ハードウェアデコーダにおいて、連続するVPDUが、複数のパイプラインステージによって同時に処理される。VPDUサイズは、ほとんどのパイプラインステージにおいてバッファサイズにほぼ比例するので、VPDUサイズを小さく保つことが重要である。ほとんどのハードウェアデコーダにおいて、VPDUサイズは、最大変換ブロック(TB)サイズに設定され得る。しかしながら、VCCにおいて、三分木(TT)および二分木(BT)パーティションは、VPDUサイズの増加につながる場合がある。

【0149】

それに加えて、ツリーノードブロックの一部がピクチャ下部または右側の境界を超える場合、ツリーノードブロックは、すべてのコーディングされたCUのすべてのサンプルがピクチャ境界の内側に配置されるまで、強制的に分割されることが留意されるべきである。

【0150】

例として、イントラサブパーティション(ISP(Intra Sub-Partition))ツールは、ルーマイントラ予測されたブロックを、垂直または水平に、ブロックサイズに応じて2または4つのサブパーティションに分割し得る。

【0151】

10

20

30

40

50

一例において、ビデオエンコーダ20のモード選択ユニット260は、本明細書で説明するパーティション分割技法の任意の組合せを実行するように構成され得る

【0152】

上記で説明したように、ビデオエンコーダ20は、(たとえば、事前に決定された)予測モードのセットから最良のまたは最適な予測モードを決定または選択するように構成される。予測モードのセットは、たとえば、イントラ予測モードおよび/またはインター予測モードを含み得る。

【0153】

イントラ予測

イントラ予測モードのセットは、たとえば、HEVCにおいて定義されているように、35の異なるイントラ予測モード、たとえば、DC(または平均)モードおよび平面モードのような無指向性モード、もしくは指向性モードを含み得、またはVVCのために定義されているように、67の異なるイントラ予測モード、たとえば、DC(または平均)モードおよび平面モードのような無指向性モード、もしくは指向性モードを含み得る。一例として、いくつかの従来の角度イントラ予測モードは、たとえば、VVCにおいて定義されているように、非正方形ブロックのための広角イントラ予測モードに適応的に置き換えられる。別の例として、DC予測のための除算演算を回避するために、非正方形ブロックに関する平均を計算するために、長辺のみが使用される。そして、平面モードのイントラ予測の結果は、位置依存イントラ予測組合せ(PDPC(position dependent intra prediction combination))法によってさらに修正され得る。

【0154】

イントラ予測ユニット254は、イントラ予測モードのセットのイントラ予測モードに従ってイントラ予測ブロック265を生成するために、同じ現在のピクチャの隣接するブロックの再構築されたサンプルを使用するように構成される。

【0155】

イントラ予測ユニット254(または一般にモード選択ユニット260)は、たとえば、ビデオデコーダ30が復号するために予測パラメータを受信および使用し得るように、イントラ予測パラメータ(または一般に、ブロックに対して選択されたイントラ予測モードを示す情報)を、符号化ピクチャデータ21に含めるために、構文要素266の形態においてエントロピー符号化ユニット270に出力するようにさらに構成される。

【0156】

インター予測

インター予測モードのセット(または可能なインター予測モード)は、利用可能な参照ピクチャ(すなわち、たとえば、DBP230内に記憶された以前の少なくとも部分的に復号されたピクチャ)および他のインター予測パラメータ、他の参照ピクチャ全体、もしくは参照ピクチャの一部、たとえば、現在のブロックの領域の周囲の探索ウィンドウ領域が、最も一致する参照ブロックを探索するために使用されるかどうか、ならびに/または、たとえば、ハーフ/セミペル、クォーターペルおよび/もしくは1/16ペル補間などのピクセル補間が適用されるかどうかに依存する。

【0157】

上記の予測モードに加えて、スキップモード、ダイレクトモード、および/または他のインター予測モードが適用され得る。

【0158】

たとえば、拡張マージ予測では、そのようなモードのマージ候補リストは、以下の5のタイプの候補、すなわち、空間的に隣接するCUからの空間的MVP、同じ場所に配置されたCUからの時間的MVP、FIFOテーブルからの履歴ベースのMVP、ペアワイズ平均MVP、およびゼロMVを順番に含むことによって構築される。また、バイラテラルマッチングベースのデコーダ側動きベクトル精密化(DMVR(decoder side motion vector refinement))が、マージモードのMVの精度を高めるために適用され得る。MVDを用いるマージモード(MMVD(Merge mode with MVD))、これは、動きベクトル差分を用いるマージモー

ドに由来する。MMVDフラグは、MMVDモードがCUに対して使用されるかどうかを指定するために、スキップフラグとマージフラグとを送信した直後にシグナリングされる。また、CUレベルの適応動きベクトル解像度(AMVR(adaptive motion vector resolution))方式が適用され得る。AMVRは、CUのMVDが異なる精度においてコーディングされることを可能にする。現在のCUに関する予測モードに応じて、現在のCUのMVDは、適応的に選択され得る。CUがマージモードにおいてコーディングされる場合、組合せインター/イントラ予測(CIIP(combined inter/intra prediction))モードが現在のCUに対して適用され得る。CIIP予測を取得するために、インター予測信号およびイントラ予測信号の加重平均化が実行される。アフィン動き補償予測では、ブロックのアフィン動き場は、2制御点の動き情報(4パラメータ)または3制御点動きベクトル(6パラメータ)によって記述される。サブブロックベースの時間的動きベクトル予測(SbTMVP(Subblock-based temporal motion vector prediction))、これは、HEVCにおける時間的動きベクトル予測(TMVP(temporal motion vector prediction))に類似しているが、現在のCU内のサブCUの動きベクトルを予測する。以前はBIOと呼ばれていた双方向オプティカルフロー(BDOF(Bi-directional optical flow))は、特に、乗算の数および乗算器のサイズの点で、はるかにより少ない計算を必要とする、より単純なバージョンである。三角形パーティションモード、このようなモードにおいて、CUは、対角分割または反対角分割のいずれかを使用して、2つの三角形形状のパーティションに均等に分割される。そのうえ、バイ予測モードは、2つの予測信号の加重平均化を可能にするために、単純な平均化を超えて拡張される。

【0159】

インター予測ユニット244は、動き推定(ME(motion estimation))ユニットと動き補償(MC(motion compensation))ユニットとを含み得る(両方とも図2には示されていない)。動き推定ユニットは、動き推定のために、ピクチャブロック203(現在のピクチャ17の現在のピクチャブロック203)と、復号されたピクチャ231、または少なくとも1つもしくは複数の以前に再構築されたブロック、たとえば、1つもしくは複数の他の/異なる以前に復号されたピクチャ231とを受信または取得するように構成され得る。たとえば、ビデオシーケンスは、現在のピクチャと、以前に復号されたピクチャ231とを含み得、または、言い換えれば、現在のピクチャおよび以前に復号されたピクチャ231は、ビデオシーケンスを形成するピクチャのシーケンスの一部であるか、またはそれを形成し得る。

【0160】

エンコーダ20は、たとえば、複数の他のピクチャの同じまたは異なるピクチャの複数の参照ブロックから参照ブロックを選択し、参照ピクチャ(または参照ピクチャインデックス)、および/または参照ブロックの位置(x、y座標)と現在のブロックの位置との間のオフセット(空間的オフセット)を、インター予測パラメータとして動き推定ユニットに提供するように構成され得る。このオフセットは、動きベクトル(MV(motion vector))とも呼ばれる。

【0161】

動き補償ユニットは、インター予測パラメータを取得、たとえば、受信し、インター予測ブロック265を取得するために、インター予測パラメータに基づいて、またはそれを使用して、インター予測を実行するように構成され得る。動き補償ユニットによって実行される動き補償は、動き推定によって決定された動き/ブロックベクトルに基づいて予測ブロックをフェッチまたは生成することを含み、場合によってはサブピクセル精度への補間を実行することを含み得る。補間フィルタリングは、既知のピクセルサンプルから追加のピクセルサンプルを生成し得、したがって、ピクチャブロックをコーディングするために使用され得る候補予測ブロックの数を増加させる可能性がある。現在のピクチャブロックのPUに関する動きベクトルを受信すると、動き補償ユニットは、参照ピクチャリストのうちの1つにおいて動きベクトルが指す予測ブロックを見つけ得る。

【0162】

動き補償ユニットは、ビデオスライスのピクチャブロックを復号する際にビデオコーダ30によって使用するために、ブロックおよびビデオスライスに関連する構文要素も生成し

10

20

30

40

50

得る。スライスおよびそれぞれの構文要素に加えて、またはその代替として、タイルグループおよび/またはタイルならびにそれぞれの構文要素が生成または使用され得る。

【0163】

エントロピーコーディング

エントロピー符号化ユニット270は、たとえば、ビデオデコーダ30が復号するためのパラメータを受信および使用し得るように、出力272を介して、たとえば、符号化ビットストリーム21の形態において出力され得る符号化ピクチャデータ21を取得するために、たとえば、エントロピー符号化アルゴリズムまたは方式(たとえば、可変長コーディング(VLC(variable length coding))方式、コンテキスト適応VLC方式(CAVLC(context adaptive VLC)、算術コーディング方式、二値化、コンテキスト適応バイナリ算術コーディング(CABAC(context adaptive binary arithmetic coding))、構文ベースの適応バイナリ算術コーディングSBAC(syntax-based context-adaptive binary arithmetic coding))、確率間隔パーティション分割エントロピー(PIPE(probability interval partitioning entropy))コーディング、または別のエントロピー符号化方法もしくは技法)を適用するか、または量子化された係数209、インター予測パラメータ、イントラ予測パラメータ、ループフィルタパラメータ、および/もしくは他の構文要素においてバイパス(無圧縮)するように構成され得る。ビットストリームは、たとえば、以下で図16においてさらに指定するような形態を有し得る。したがって、この図に関連して説明する実施形態は、ここで説明するビットストリーム21内にも包含されると見なされる。さらに、本明細書において言及されるビットストリームの任意の構造は、この実施形態の意味においてビットストリーム21として提供され得る。符号化ビットストリーム21は、ビデオデコーダ30に送信されるか、またはビデオデコーダ30による後の送信または検索のためにメモリ内に記憶され得る。

10

20

【0164】

ビデオエンコーダ20の他の構造的バリエーションが、ビデオストリームを符号化するために使用され得る。たとえば、非変換ベースのエンコーダ20は、特定のブロックまたはフレームについて、変換処理ユニット206なしで、残差信号を直接量子化することができる。別の実装形態において、エンコーダ20は、単一のユニットに結合された量子化ユニット208および逆量子化ユニット210を有することができる。

【0165】

デコーダおよび復号方法

図3は、この本開示の技法を実装するように構成され得るビデオデコーダ30の一例を示す。ビデオデコーダ30は、復号されたピクチャ331を取得するために、たとえば、エンコーダ20によって符号化された符号化ピクチャデータ21(たとえば、符号化ビットストリーム21)を受信するように構成され得る。符号化ピクチャデータまたはビットストリームは、符号化ピクチャデータ、たとえば、符号化ビデオスライスのピクチャブロック(および/またはタイルグループまたはタイル)を表すデータを復号するための情報と、関連する構文要素とを含む。

30

【0166】

図3の例において、デコーダ30は、エントロピー復号ユニット304と、逆量子化ユニット310と、逆変換処理ユニット312と、再構築ユニット314(たとえば、アナログ加算器314)と、ループフィルタ320と、復号ピクチャバッファ(DBP)330と、モード適用ユニット360と、インター予測ユニット344と、イントラ予測ユニット354とを備える。インター予測ユニット344は、動き補償ユニットであるか、またはそれを含み得る。ビデオデコーダ30は、いくつかの例において、図2からのビデオエンコーダ100に関して説明した符号化パスと概ね逆の復号パスを実行し得る。

40

【0167】

エンコーダ20に関して説明したように、逆量子化ユニット210、逆変換処理ユニット212、再構築ユニット214、ループフィルタ220、復号ピクチャバッファ(DPB)230、インター予測ユニット344、およびイントラ予測ユニット354は、ビデオエンコーダ20の「ピ

50

ルトインデコーダ」を形成するものとも呼ばれる。したがって、逆量子化ユニット310は、逆量子化ユニット110と機能において同一であり得、逆変換処理ユニット312は、逆変換処理ユニット212と機能において同一であり得、再構築ユニット314は、再構築ユニット214と機能において同一であり得、ループフィルタ320は、ループフィルタ220と機能において同一であり得、復号ピクチャバッファ330は、復号ピクチャバッファ230と機能において同一であり得る。したがって、ビデオ20エンコーダのそれぞれのユニットおよび機能について提供した説明は、ビデオデコーダ30のそれぞれのユニットおよび機能に対応して適用される。

【0168】

エントロピー復号

エントロピー復号ユニット304は、ビットストリーム21(または一般に符号化ピクチャデータ21)を解析し、たとえば、量子化係数309および/または復号されたコーディングパラメータ(図3には示さず)、たとえば、インター予測パラメータ(たとえば、参照ピクチャインデックスおよび動きベクトル)、イントラ予測パラメータ(たとえば、イントラ予測モードまたはインデックス)、変換パラメータ、量子化パラメータ、ループフィルタパラメータ、および/または他の構文要素のうちのいずれかまたはすべてを取得するために、たとえば、符号化ピクチャデータ21に対してエントロピー復号を実行するように構成され得る。エントロピー復号ユニット304は、エンコーダ20のエントロピー符号化ユニット270に関して説明したような符号化方式に対応する復号アルゴリズムまたは方式を適用するように構成され得る。エントロピー復号ユニット304は、インター予測パラメータ、イントラ予測パラメータ、および/または他の構文要素をモード適用ユニット360に提供し、他のパラメータをデコーダ30の他のユニットに提供するようにさらに構成され得る。ビデオデコーダ30は、ビデオスライスレベルおよび/またはビデオブロックレベルにおいて構文要素を受信し得る。スライスおよびそれぞれの構文要素に加えて、またはその代替として、タイルグループおよび/またはタイルならびにそれぞれの構文要素が受信および/または使用され得る。

【0169】

逆量子化

逆量子化ユニット310は、(たとえば、エントロピー復号ユニット304によって、たとえば、解析および/または復号することによって)符号化ピクチャデータ21からの量子化パラメータ(QP(quantization parameter))(または一般に、逆量子化に関連する情報)と量子化係数とを受信し、変換係数311とも呼ばれる場合がある脱量子化係数311を取得するために、量子化パラメータに基づいて、復号された量子化係数309に対して逆量子化を適用するように構成され得る。逆量子化プロセスは、量子化の程度と、同様に、適用されるべき逆量子化の程度とを決定するために、ビデオスライス(またはタイルもしくはタイルグループ)内の各ビデオブロックに対してビデオエンコーダ20によって決定された量子化パラメータを使用することを含む。

【0170】

逆変換

逆変換処理ユニット312は、変換係数311とも呼ばれる脱量子化係数311を受信し、サンプルドメインにおいて再構築された残差ブロック213を取得するために、脱量子化係数311に変換を適用するように構成され得る。再構築された残差ブロック213は、変換ブロック313とも呼ばれる場合もある。変換は、逆変換、たとえば、逆DCT、逆DST、逆整数変換、または概念的に類似した逆変換プロセスであり得る。逆変換処理ユニット312は、脱量子化係数311に対して適用されるべき変換を決定するために、(たとえば、エントロピー復号ユニット304によって、たとえば、解析および/または復号することによって)符号化ピクチャデータ21からの変換パラメータまたは対応する情報を受信するようにさらに構成され得る。

【0171】

再構築

再構築ユニット314(たとえば、加算器またはアナログ加算器314)は、たとえば、再構築された残差ブロック313のサンプル値と予測ブロック365のサンプル値とを加算することによって、サンプルドメインにおいて再構築されたブロック315を取得するために、再構築された残差ブロック313を予測ブロック365に加算するように構成され得る。

【0172】

フィルタリング

(コーディンググループ内またはコーディンググループ後のいずれかの)ループフィルタユニット320は、たとえば、ピクセル遷移を滑らかにするか、またはビデオ品質を他の方法で改善するために、フィルタリングされたブロック321を取得するために、再構築されたブロック315をフィルタリングするように構成され得る。ループフィルタユニット320は、デブロッキングフィルタ、サンプル適応オフセット(SAO)フィルタ、または1つもしくは複数の他のフィルタ、たとえば、適応ループフィルタ(ALF)、ノイズ抑制フィルタ(NSF)、またはそれらの任意の組合せなどの、1つまたは複数のループフィルタを備え得る。一例において、ループフィルタユニット220は、デブロッキングフィルタ、SAOフィルタ、およびALFフィルタを備え得る。フィルタリングプロセスの順序は、デブロッキングフィルタ、SAO、およびALFであり得る。別の例において、クロマスケーリングによるルママッピング(LMCS)(すなわち、適応ループ内リシェーパ)と呼ばれるプロセスが追加される。このプロセスは、デブロッキングの前に実行される。別の例において、デブロッキングフィルタプロセスは、内部サブブロックエッジ、たとえば、アフィンサブブロックエッジ、AT MVPサブブロックエッジ、サブブロック変換(SBT)エッジ、およびイントラサブパーティション(ISP)エッジにも適用され得る。ループフィルタユニット320は、図3においてループ内フィルタであるように示されているが、他の構成では、ループフィルタユニット320は、ポストループフィルタとして実装され得る。

【0173】

JVET-P0080およびJVET-00630は、クロスコンポーネントALFフィルタと呼ばれ、本明細書ではCC-ALFまたはCCALFとも呼ばれる新しいループ内フィルタを提案している。CC-ALFは、適応ループフィルタプロセスの一部として動作し、各クロマ成分を精緻化するためにルマサンプル値を利用する(すなわち、第1のクロマ成分などのCrまたはCb成分は、すなわちCb成分であり、第2のクロマ成分は、すなわちCr成分である)。CC-ALFは、クロマ成分のクロマサンプルごとにルマ成分に対してダイヤモンド形フィルタを適用することによって動作し、次いで、出力されたフィルタリングされた値は、クロマALFプロセスの出力に対する補正として使用される。

【0174】

クロスコンポーネント適応ループフィルタ(CC-ALF)は、それぞれ、ループフィルタおよび後処理ステップとして使用され得る。

【0175】

復号ピクチャバッファ

次いで、ピクチャの復号されたビデオブロック321は、復号ピクチャバッファ330内に記憶され、復号ピクチャバッファ330は、復号されたピクチャ331を、他のピクチャのためのその後の動き補償のため、および/または表示をそれぞれ出力するための参照ピクチャとして記憶する。

【0176】

デコーダ30は、ユーザに対して提示または表示するために、たとえば、出力312を介して、復号されたピクチャ311を出力するように構成され得る。

【0177】

予測

インター予測ユニット344は、インター予測ユニット244(特に、動き補償ユニット)と同一であり得、イントラ予測ユニット354は、機能においてインター予測ユニット254と同一であり得、(たとえば、エントロピー復号ユニット304によって、たとえば、解析および/または復号することによって)符号化ピクチャデータ21からの受信されたパーティショ

10

20

30

40

50

ン分割および/もしくは予測パラメータまたはそれぞれの情報に基づいて、分割またはパーティション分割の決定および予測を実行する。モード適用ユニット360は、予測ブロック365を取得するために、再構築されたピクチャ、ブロック、またはそれぞれのサンプル(フィルタリングされたまたはフィルタリングされていない)に基づいて、ブロックごとに予測(イントラまたはインター予測)を実行するように構成され得る。

【0178】

ビデオスライスがイントラコーディングされた(I)スライスとしてコーディングされる場合、モード適用ユニット360のイントラ予測ユニット354は、現在のピクチャの以前に復号されブロックからのシグナリングされたイントラ予測モードおよびデータに基づいて、現在のビデオスライスのピクチャブロックに関する予測ブロック365を生成するように構成される。ビデオピクチャがインターコーディングされた(すなわち、BまたはP)スライスとしてコーディングされる場合、モード適用ユニット360のインター予測ユニット344(たとえば、動き補償ユニット)は、エントロピー復号ユニット304から受信された動きベクトルおよび他の構文要素に基づいて、現在のビデオスライスのビデオブロックに関する予測ブロック365を生成するように構成される。インター予測の場合、予測ブロックは、参照ピクチャリストのうちの1つの中の参照ピクチャリストのうちの1つから生成され得る。ビデオデコーダ30は、DPB330内に記憶された参照ピクチャに基づいて、デフォルトの構築技術を使用して、参照フレームリスト、リスト0およびリスト1を構築し得る。同じことまたは類似のことが、スライス(たとえば、ビデオスライス)に加えて、またはその代わりに、タイルグループ(たとえば、ビデオタイルグループ)および/またはタイル(たとえば、ビデオタイル)を使用する実施形態に対してまたはそれによって適用され得、たとえば、ビデオは、I、P、またはBタイルグループおよび/またはタイルを使用してコーディングされ得る。

【0179】

モード適用ユニット360は、動きベクトルまたは関連情報および他の構文要素を解析することによって現在のビデオスライスのビデオブロックに関する予測情報を決定するように構成され得、復号されている現在のビデオブロックに関する予測ブロックを生成するために予測情報を使用する。たとえば、モード適用ユニット360は、ビデオスライスのビデオブロックをコーディングするために使用される予測モード(たとえば、イントラまたはインター予測)と、インター予測スライスタイプ(たとえば、Bスライス、Pスライス、またはGPBスライス)、スライスに関する参照ピクチャリストのうちの1つまたは複数に関する構築状態と、スライスの各々のインター符号化されたビデオブロックに関する動きベクトルと、スライスの各々のインターコーディングされたビデオブロックに関するインター予測ステータスと、現在のビデオスライス内のビデオブロックを復号するための他の情報とを決定するために、受信された構文要素のうちのいくつかを使用する。同じことまたは類似のことが、スライス(たとえば、ビデオスライス)に加えて、またはその代わりに、タイルグループ(たとえば、ビデオタイルグループ)および/またはタイル(たとえば、ビデオタイル)を使用する実施形態に対してまたはそれによって適用され得、たとえば、ビデオは、I、P、もしくはBタイルグループおよび/またはタイルを使用してコーディングされ得る。

【0180】

図3において示すようなビデオデコーダ30の実施形態は、スライス(ビデオスライスとも呼ばれる)を使用することによって、ピクチャをパーティション分割および/または復号するように構成され得、ピクチャは、(典型的には重複しない)1つまたは複数のスライスを使用してパーティション分割または復号され得、各スライスは、1つもしくは複数のブロック(たとえば、CTU)、またはブロック(たとえば、タイル(H.265/HEVCおよびVVC)またはブリック(VVC))の1つもしくは複数のグループを備え得る。

【0181】

図3において示すようなビデオデコーダ30の実施形態は、スライス/タイルグループ(ビデオタイルグループとも呼ばれる)および/またはタイル(ビデオタイルとも呼ばれる)を使用することによって、ピクチャをパーティション分割および/または復号するように構成さ

れ得、ピクチャは、(典型的には重複しない)1つまたは複数のスライス/タイルグループを使用してパーティション分割または復号され得、各スライス/タイルグループは、たとえば、1つもしくは複数のブロック(たとえば、CTU)または1つもしくは複数のタイルを備え得、各タイルは、たとえば、矩形形状であり得、1つまたは複数のブロック(たとえば、CTU)、たとえば、完全なまたは断片のブロックを備え得る。

【0182】

ビデオデコーダ30の他のバリエーションが、符号化ピクチャデータ21を復号するために使用され得る。たとえば、デコーダ30は、ループフィルタリングユニット320なしで、出力ビデオストリームを生成することができる。たとえば、非変換ベースのデコーダ30は、特定のブロックまたはフレームについて、逆変換処理ユニット312なしで、残差信号を直接逆量子化することができる。別の実装形態において、ビデオデコーダ30は、単一のユニットに結合された逆量子化ユニット310および逆変換処理ユニット312を有することができる。

10

【0183】

エンコーダ20およびデコーダ30において、現在のステップの処理結果は、さらに処理され、次いで、次のステップに出力され得るものと理解されるべきである。たとえば、補間フィルタリング、動きベクトル導出、またはループフィルタリングの後に、補間フィルタリング、動きベクトル導出、またはループフィルタリングの処理結果に対して、クリップまたはシフトなどのさらなる演算が実行され得る。

【0184】

20

現在のブロックの導出された動きベクトル(限定はしないが、アフィンモードの制御点動きベクトル、アフィンモード、平面モード、ATMVPモードにおけるサブブロック動きベクトル、時間的動きベクトルなどを含む)に対して、さらなる演算が適用され得ることが留意されるべきである。たとえば、動きベクトルの値は、その表現ビットに従って、事前定義された範囲に制約される。動きベクトルの表現ビットがbitDepthである場合、範囲は、 $-2^{(bitDepth-1)} \sim 2^{(bitDepth-1)}-1$ であり、ここで、「 $\wedge$ 」は、べき乗を意味する。たとえば、bitDepthが16に設定されている場合、範囲は、 $-32768 \sim 32767$ であり、bitDepthが18に設定されている場合、範囲は、 $-131072 \sim 131071$ である。たとえば、導出された動きベクトル(たとえば、1つの 8×8 ブロック内の4つの 4×4 サブブロックのMV)の値は、4つの 4×4 サブブロックMVの整数部分間の最大差が1ピクセル以下など、Nピクセル以下であるように制約される。ここで、bitDepthに従って動きベクトルを制約するための2つの方法を提供する。

30

【0185】

図4は、本開示の一実施形態によるビデオコーディングデバイス400の概略図である。ビデオコーディングデバイス400は、本明細書において説明する開示された実施形態を実装するのに適している。一実施形態において、ビデオコーディングデバイス400は、図1Aのビデオデコーダ30などのデコーダ、または図1Aのビデオエンコーダ20などのエンコーダであり得る。

【0186】

ビデオコーディングデバイス400は、データを受信するための入力ポート410(または入力ポート410)および受信機ユニット(Rx)420と、データを処理するためのプロセッサ、論理ユニット、または中央処理装置(CPU)430と、データを送信するための送信機ユニット(Tx)440および出口ポート450(または出力ポート450)と、データを記憶するためのメモリ460とを備える。ビデオコーディングデバイス400は、光信号または電気信号の出口または入口のための、入力ポート410、受信機ユニット420、送信機ユニット440、および出口ポート450に結合された光-電気(OE)構成要素および電気-光(E0)構成要素も備え得る。

40

【0187】

プロセッサ430は、ハードウェアおよびソフトウェアによって実装される。プロセッサ430は、1つまたは複数のCPUチップ、コア(マルチコアプロセッサとして)、FPGA、ASIC、およびDSPとして実装され得る。プロセッサ430は、入力ポート410、受信機ユニッ

50

ト420、送信機ユニット440、出口ポート450、およびメモリ460と通信している。プロセッサ430は、コーディングモジュール470を備える。コーディングモジュール470は、上記で説明した開示された実施形態を実装する。たとえば、コーディングモジュール470は、様々なコーディング動作を実装、処理、準備、または提供する。したがって、コーディングモジュール470の包含は、ビデオコーディングデバイス400の機能に実質的な改善を提供し、ビデオコーディングデバイス400の異なる状態への変換をもたらす。代替的には、コーディングモジュール470は、メモリ460内に記憶され、プロセッサ430によって実行される命令として実装される。

【0188】

メモリ460は、1つまたは複数のディスク、テープドライブ、およびソリッドステートドライブを備え得、プログラムが実行のために選択されたときにそのようなプログラムを記憶し、プログラム実行中に読み出される命令およびデータを記憶するために、オーバフローデータ記憶デバイスとして使用され得る。メモリ460は、たとえば、たとえば、揮発性および/または不揮発性であり得、読み取り専用メモリ(ROM(read-only memory))、ランダムアクセスメモリ(RAM(random access memory))、三元連想メモリ(TCAM(ternary content-addressable memory))、および/またはスタティックランダムアクセスメモリ(SRAM(static random-access memory))であり得る。

【0189】

図5は、例示的な実施形態による、図1からのソースデバイス12および宛先デバイス14のいずれかまたは両方として使用され得る装置500の簡略化されたブロック図である。

【0190】

装置500内のプロセッサ502は、中央処理装置であり得る。代替的には、プロセッサ502は、現在存在する、または今後開発される情報を操作または処理することができる、任意の他のタイプのデバイス、または複数のデバイスであり得る。代替的には、開示されている実装形態は、示されているような単一のプロセッサ、たとえば、プロセッサ202を用いて実施され得るが、速度および効率における利点は、2つ以上のプロセッサを使用して達成され得る。

【0191】

装置500内のメモリ504は、一実装形態において、読み取り専用メモリ(ROM)デバイスまたはランダムアクセスメモリ(RAM)デバイスであり得る。任意の他の適切なタイプの記憶デバイスが、メモリ504として使用され得る。メモリ504は、バス512を使用してプロセッサ502によってアクセスされるコードおよびデータ506を含むことができる。メモリ504は、オペレーティングシステム508と、アプリケーションプログラム510とをさらに含むことができ、アプリケーションプログラム510は、プロセッサ502が本明細書で説明する方法を実行することを可能にする少なくとも1つのプログラムを含む。たとえば、アプリケーションプログラム510は、アプリケーション1からNを含むことができ、アプリケーション1からNは、本明細書で説明する方法を実行するビデオコーディングアプリケーションをさらに含む。

【0192】

装置500は、ディスプレイ518などの1つまたは複数の出力デバイスを含むこともできる。ディスプレイ518は、一例において、タッチ入力を感じ取るように動作可能なタッチ感知要素とディスプレイを組み合わせたタッチ感知ディスプレイであり得る。ディスプレイ518は、バス512を介してプロセッサ502に結合され得る。

【0193】

ここでは単一のバスとして示されているが、装置500のバス512は、複数のバスから構成され得る。さらに、二次記憶装置514は、装置500の他の構成要素に直接結合されるか、またはネットワークを介してアクセスされ得、メモリカードなどの単一の統合ユニット、または複数のメモリカードなどの複数のユニットを備えることができる。したがって、装置500は、多種多様な構成において実装され得る。

【0194】

本開示の実施形態の詳細な説明

ビデオコーディングは、色空間と色フォーマットとに基づいて実行され得る。たとえば、カラービデオは、色を効率的に表現するために様々な色空間が使用されるマルチメディアシステムにおいて重要な役割を果たす。色空間は、複数の成分を使用して数値を用いて色を指定する。一般的な色空間は、RGB色空間であり、色は、3つの原色成分値(すなわち、赤色、緑色、および青色)の組合せとして表現される。カラービデオ圧縮については、A. FordおよびA. Roberts、「Colour space conversions」、University of Westminster、ロンドン、Tech. Rep、1998年8月において記載されているように、YCbCr色空間が広く使用されている。

【0195】

YCbCrは、線形変換を介してRGB色空間から容易にコンバートされ得、異なる成分間の冗長性、すなわちクロスコンポーネントの冗長性は、YCbCr色空間において大幅に削減される。YCbCrの1つの利点は、Y信号がルミナンス情報を伝達するので、白黒テレビとの後方互換性である。それに加えて、4:2:0クロマサンプリングフォーマットにおいてCb成分およびCr成分をサブサンプリングすることによって、RGB色空間におけるサブサンプリングよりも大幅に少ない主観的な影響で、クロミナンス帯域幅が削減され得る。これらの利点のために、YCbCrは、ビデオ圧縮における主要な色空間であった。ビデオ圧縮において使用されるYCoCgなどの他の色空間も存在する。本開示において、使用される実際の色空間に関係なく、ルマ(またはLもしくはY)および2つのクロマ(CbおよびCr)が、ビデオ圧縮方式における3つの色成分を表すために使用される。

【0196】

たとえば、クロマフォーマットサンプリング構造は、4:2:0サンプリングであるとき、2つのクロマ配列の各々は、ルマ配列の半分の高さで半分の幅とを有する。ピクチャ内のルマサンプルおよびクロマサンプルの公称の垂直および水平の相対位置は、図9Aにおいて示されている。図9Bは、4:2:0サンプリングの一例を示す。図9Bは、同じ場所に配置されたルマブロックおよびクロマブロックの一例を示す。ビデオフォーマットがYUV4:2:0である場合、1つの16×16ルマブロックと、2つの8×8クロマブロックとが存在する。

【0197】

具体的には、コーディングブロックまたは変換ブロックは、ルマブロックと2つのクロマブロックとを含む。

【0198】

図示のように、ルマブロックは、クロマブロックの4倍のサンプルを含む。具体的には、クロマブロックは、N個のサンプル×N個のサンプルを含み、ルマブロックは、2N個のサンプル×2N個のサンプルを含む。したがって、ルマブロックは、クロマブロックの4倍の解像度である。たとえば、YUV4:2:0フォーマットが使用される場合、ルマサンプルは、4倍(たとえば、幅2倍、高さ2倍)ダウンサンプリングされ得る。YUVは、ルマ成分Yならびに2つのクロミナンス成分UおよびVに関して色空間を用いる色符号化システムである。

【0199】

ピクチャヘッダ

ピクチャヘッダの概念が、(JVET-P1006、P0095、P0120、P0239において提示されているように)VVC規格において新たに導入された。ピクチャヘッダの構文について、JVET-P2001-VEにおけるセクション7.3.2.6を参照されたい。

【0200】

現在のVVCドラフトにおいて、ピクチャの最初のVCL NALユニットとしてピクチャごとに1回送信されるように必須ピクチャヘッダの概念が提案されている。現在のVVCドラフトはまた、スライスヘッダ内に現在存在する構文要素のうちのいくつかをこのピクチャヘッダに移動する。機能的にピクチャごとに1回送信されることのみを必要とする構文要素は、所与のピクチャについて複数回送信される代わりに、ピクチャヘッダに移動され、たとえば、スライスヘッダ内の構文要素は、スライスごとに1回送信される。スライスヘッ

10

20

30

40

50

ダ処理に必要な計算が全体的なスループットに対する制限要因になる可能性があるので、スライスヘッダから構文要素を移動することによって見られる利益が存在する。

【0201】

適応ループフィルタ(ALF)の場合、以下の構文要素(ALF関連構文要素)がピクチャヘッダ内に導入された。

【0202】

【表1】

7.3.2.6 ピクチャヘッダ RBSP 構文

picture_header_rbsp() {	記述子
.....	
if(sps_alf_enabled_flag) {	
pic_alf_enabled_present_flag	u(1)
if(pic_alf_enabled_present_flag) {	
pic_alf_enabled_flag	u(1)
if(pic_alf_enabled_flag) {	
pic_num_alf_aps_ids_luma	u(3)
for(i = 0; i < pic_num_alf_aps_ids_luma; i++)	
pic_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
pic_alf_chroma_idc	u(2)
if(pic_alf_chroma_idc)	
pic_alf_aps_id_chroma	u(3)
}	
}	
}	
.....	

【0203】

ここで、構文要素は、**pic_alf_enabled_present_flag**、**pic_alf_enabled_flag**、**pic_num_alf_aps_ids_luma**、**pic_alf_aps_id_luma[i]**、**pic_alf_chroma_idc**、**pic_alf_aps_id_chroma**である。これらの構文要素は、ピクチャヘッダにおいて提供され、その存在は、以前にまたは他の方法でシグナリングされた他の構文要素に依存する可能性がある。たとえば、**sps_alf_enabled_flag**および**ChromaArrayType**がそのような他の構文要素である。

【0204】

以下において、表中の記述子で示されている、および/または太字で提供されている任意の構文要素は、現在の構文構造においてシグナリングまたは提供される構文要素である。

【0205】

スライスヘッダにおいて、以下の構文変更がALFに対して導入された。

【0206】

10

20

30

40

50

【表 2】

7.3.7.1 一般的なスライスヘッダ構文

slice_header() {	記述子
.....	
if(sps_alf_enabled_flag && !pic_alf_enabled_present_flag) {	
slice_alf_enabled_flag	u(1)
if(slice_alf_enabled_flag) {	
slice_num_alf_aps_ids_luma	u(3)
for(i = 0; i < slice_num_alf_aps_ids_luma; i++)	
slice_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
slice_alf_chroma_idc	u(2)
if(slice_alf_chroma_idc)	
slice_alf_aps_id_chroma	u(3)
}	
}	
.....	

10

20

【0207】

ALFピクチャヘッダエントリおよびスライスヘッダエントリのセマンティクスは、以下のとおりである。

【0208】

1に等しいpic_alf_enabled_present_flagは、pic_alf_enabled_flag、pic_num_alf_aps_ids_luma、pic_alf_aps_id_luma[i]、pic_alf_chroma_idc、およびpic_alf_aps_id_chromaがPH内に存在することを指定する。0に等しいpic_alf_enabled_present_flagは、pic_alf_enabled_flag、pic_num_alf_aps_ids_luma、pic_alf_aps_id_luma[i]、pic_alf_chroma_idc、およびpic_alf_aps_id_chromaがPH内に存在しないことを指定する。pic_alf_enabled_present_flagが存在しない場合、0に等しいと推定される。

30

【0209】

1に等しいpic_alf_enabled_flagは、適応ループフィルタがPHに関連付けられたすべてのスライスに対して有効にされており、スライス内のY、Cb、またはCr色成分に適用され得ることを指定する。0に等しいpic_alf_enabled_flagは、適応ループフィルタがPHに関連付けられた1つ、または複数、またはすべてのスライスに対して無効にされ得ることを指定する。存在しない場合、pic_alf_enabled_flagは、0に等しいと推定される。

【0210】

pic_num_alf_aps_ids_lumaは、PHに関連付けられたスライスが参照するALF APSの数を指定する。

40

【0211】

pic_alf_aps_id_luma[i]は、PHに関連付けられたスライスのルマ成分が参照するi番目のALF APSのadaptation_parameter_set_idを指定する。

【0212】

ALF_APSに等しいaps_params_typeとpic_alf_aps_id_luma[i]に等しいadaptation_parameter_set_idとを有するAPS NALユニットのalf_luma_filter_signal_flagの値は、1に等しいべきである。

【0213】

0に等しいpic_alf_chroma_idcは、適応ループフィルタがCbおよびCr色成分に適用されないことを指定する。1に等しいpic_alf_chroma_idcは、適応ループフィルタがCb色

50

成分に適用されることを示す。2に等しいpic_alf_chroma_idcは、適応ループフィルタがCr色成分に適用されることを示す。3に等しいpic_alf_chroma_idcは、適応ループフィルタがCbおよびCr色成分に適用されることを示す。pic_alf_chroma_idcが存在しない場合、0に等しいと推定される。

【0 2 1 4】

pic_alf_aps_id_chromaは、PHに関連付けられたスライスのクロマ成分が参照するALF APSのadaptation_parameter_set_idを指定する。

【0 2 1 5】

ALF_APSに等しいaps_params_typeとpic_alf_aps_id_chromaに等しいadaptation_parameter_set_idとを有するAPS NALユニットのalf_chroma_filter_signal_flagの値は、1に等しいべきである。

10

【0 2 1 6】

1に等しいslice_alf_enabled_flagは、適応ループフィルタが有効にされており、スライス内のY、Cb、またはCr色成分に適用され得ることを指定する。0に等しいslice_alf_enabled_flagは、適応ループフィルタがスライス内のすべての色成分に対して無効にされていることを指定する。存在しない場合、slice_alf_enabled_flagの値は、pic_alf_enabled_flagに等しいと推定される。

【0 2 1 7】

slice_num_alf_aps_ids_lumaは、スライスが参照するALF APSの数を指定する。slice_alf_enabled_flagが1に等しく、slice_num_alf_aps_ids_lumaが存在しない場合、slice_num_alf_aps_ids_lumaの値は、pic_num_alf_aps_ids_lumaの値に等しいと推定される。

20

【0 2 1 8】

slice_alf_aps_id_luma[i]は、スライスのルマ成分が参照するi番目のALF APSのadaptation_parameter_set_idを指定する。ALF_APSに等しいaps_params_typeとslice_alf_aps_id_luma[i]に等しいadaptation_parameter_set_idとを有するAPS NALユニットのTemporalIdは、コーディングされたスライスNALユニットのTemporalId以下であるべきである。slice_alf_enabled_flagが1に等しく、slice_alf_aps_id_luma[i]が存在しない場合、slice_alf_aps_id_luma[i]の値は、pic_alf_aps_id_luma[i]の値に等しいと推定される。

30

【0 2 1 9】

ALF_APSに等しいaps_params_typeとslice_alf_aps_id_luma[i]に等しいadaptation_parameter_set_idとを有するAPS NALユニットのalf_luma_filter_signal_flagの値は、1に等しいべきである。

【0 2 2 0】

0に等しいslice_alf_chroma_idcは、適応ループフィルタがCbおよびCr色成分に適用されないことを指定する。1に等しいslice_alf_chroma_idcは、適応ループフィルタがCb色成分に適用されることを示す。2に等しいslice_alf_chroma_idcは、適応ループフィルタがCr色成分に適用されることを示す。3に等しいslice_alf_chroma_idcは、適応ループフィルタがCbおよびCr色成分に適用されることを示す。slice_alf_chroma_idcが存在しない場合、pic_alf_chroma_idcに等しいと推定される。

40

【0 2 2 1】

slice_alf_aps_id_chromaは、スライスのクロマ成分が参照するALF APSのadaptation_parameter_set_idを指定する。ALF_APSに等しいaps_params_typeとslice_alf_aps_id_chromaに等しいadaptation_parameter_set_idとを有するAPS NALユニットのTemporalIdは、コーディングされたスライスNALユニットのTemporalId以下であるべきである。slice_alf_enabled_flagが1に等しく、slice_alf_aps_id_chromaが存在しない場合、slice_alf_aps_id_chromaの値は、pic_alf_aps_id_chromaの値に等しいと推定される。

【0 2 2 2】

50

ALF_APSに等しいaps_params_typeとslice_alf_aps_id_chromaに等しいadaptation_parameter_set_idとを有するAPS NALユニットのalf_chroma_filter_signal_flagの値は、1に等しいべきである。

【0223】

上記で説明したように、すべてのスライスが同じALFフィルタリングデータを有する場合、スライスヘッダの各々において別々にALFフィルタリングデータを送信する代わりに、すべてのスライスにわたって共通のALFフィルタリングデータが、ピクチャヘッダにおいて1回だけ送信され、結果として、すべてのスライスが、ピクチャヘッダからのALFフィルタリングデータを継承する。このようにして、スライスヘッダのオーバーヘッドが(ビット数の観点から)削減される。

【0224】

本開示は、ピクチャの各スライスヘッダにおいてシグナリングされ、ピクチャヘッダにおいて定義されるCCALFのすべての共通の構文要素を収集する。本開示は、ALFシグナリングの同じ原理をクロスコンポーネントALF(CCALF)にも拡張しようとする。現在、CCALFについて、スライスヘッダの各々は、従来の方法において以下の情報を送信しなければならない。

【0225】

【表3】

7.3.7.1 一般的なスライスヘッダ構文

slice_header() {	記述子
.....	
if(ChromaArrayType != 0)	
slice_cross_component_alf_cb_enabled_flag	u(1)
if(slice_cross_component_alf_cb_enabled_flag) {	
slice_cross_component_alf_cb_aps_id	u(3)
slice_cross_component_cb_filters_signalled_minus1	ue(v)
}	
if(ChromaArrayType != 0)	
slice_cross_component_alf_cr_enabled_flag	u(1)
if(slice_cross_component_alf_cr_enabled_flag) {	
slice_cross_component_alf_cr_aps_id	u(3)
slice_cross_component_cr_filters_signalled_minus1	ue(v)
}	
.....	

【0226】

したがって、各スライスが、情報が所与のピクチャ内のすべてのスライスにわたって同じであっても、上記のすべての構文要素を送信しなければならない。

【0227】

したがって、スライスヘッダのオーバーヘッドを低減するために、本発明は、CCALFについてもピクチャヘッダエントリを定義する。

【0228】

1.1 本開示によって解決されるべき技術的問題

本開示は、スライスオーバーヘッドを低減するために、CCALFのためのピクチャヘッダエントリを導入する。図7において示すように、スライス1からスライスNは、同じCCALFフィルタ情報7001を含む。したがって、各スライスヘッダは、同じデータを送信しなけれ

ばならず、冗長性とスライスビットシグナリングオーバーヘッドとを結果として生じる。図8に示すように、CCALFデータにおけるこの冗長性を除去するために、共通のCCALFデータ(すなわちCCALF関連情報も)を定義するCCALFのためのピクチャヘッダエントリ8001が導入され、次いで、すべてのスライスは、この共通情報8002を継承することができる。したがって、これは、シグナリングにおける冗長性を除去し、スライスヘッダの解析オーバーヘッドを低減する。

【0229】

1.2 本開示の技術的実装の実施形態

「代替」として開示されている以下の実施形態の各々は、他の(代替的な)実施形態のいずれかと組み合わせて提供され得、具体的には、上記の図において言及されているエンコーダおよび/またはデコーダなどの上記で説明したデバイスのいずれかを使用して実装され得ることに留意されたい。

【0230】

1.2.1 代替実施形態1

第1のステップにおいて、CCALFが有効にされているかどうかを制御する(以下、たとえば、`sps_ccalf_enabled_flag`で示される)「第2の構文要素」と呼ばれる新しいシーケンスパラメータセット(SPS)構文要素が導入される。構文は、以下に示すとおりである。第2の構文要素(`sps_ccalf_enabled_flag`)は、ALF動作とCCALF動作とを完全に分離し、したがって、ALFおよびCCALFがシーケンスレベルにおいて別々にオンまたはオフにされることを可能にする。

【0231】

【表4】

7.3.2.3 シーケンスパラメータセット RBSP 構文

seq_parameter_set_rbsp() {	記述子
.....	
<code>sps_alf_enabled_flag</code>	u(1)
<code>sps_ccalf_enabled_flag</code>	u(1)
.....	

【0232】

ここで、`sps_alf_enabled_flag`は、SPSレベル構文において同様に提供される第1の構文要素の一例である。第1および第2の構文要素は、上記の表において例示的に提供されているように、互いに独立してシグナリングされ得る。しかしながら、たとえば、以下の代替実施形態5において提供されているように、第2の構文要素はまた、たとえば、第1の構文要素が有するまたは取る値に応じてシグナリングされ得る。

【0233】

新しいピクチャヘッダエントリ(太字および斜体においてマークされている)は、以下に示すとおりである。

【0234】

10

20

30

40

50

【表 5】

7.3.2.6 ピクチャヘッダ RBSP 構文

picture_header_rbsp() {	記述子
.....	
if(sps_alf_enabled_flag) {	
pic_alf_enabled_present_flag	u(1)
if(pic_alf_enabled_present_flag) {	
pic_alf_enabled_flag	u(1)
if(pic_alf_enabled_flag) {	
pic_num_alf_aps_ids_luma	u(3)
for(i = 0; i < pic_num_alf_aps_ids_luma; i++)	
pic_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
pic_alf_chroma_idc	u(2)
if(pic_alf_chroma_idc)	
pic_alf_aps_id_chroma	u(3)
}	
}	
}	
if(sps_ccalf_enabled_flag) {	
pic_ccalf_enabled_present_flag	u(1)
if(pic_ccalf_enabled_present_flag) {	
pic_ccalf_enabled_flag	u(1)
if(pic_ccalf_enabled_flag) {	
if(ChromaArrayType != 0)	
pic_cross_component_alf_cb_enabled_flag	u(1)
if(pic_cross_component_alf_cb_enabled_flag) {	
pic_cross_component_alf_cb_aps_id	u(3)
pic_cross_component_cb_filters_signalled_minus1	ue(v)
}	
if(ChromaArrayType != 0)	
pic_cross_component_alf_cr_enabled_flag	u(1)
if(pic_cross_component_alf_cr_enabled_flag) {	
pic_cross_component_alf_cr_aps_id	u(3)
pic_cross_component_cr_filters_signalled_minus1	ue(v)
}	
}	
}	
}	
.....	

【0 2 3 5】

ここで、さらなる構文要素がピクチャヘッダにおいて導入され、これらのさらなる構文要素は、第1および/または第2の構文要素が特定の値を取る場合にのみ存在し得る。これは、第1の構文要素および/または第2の構文要素の値に応じて、「if」構文を用いて示される。

【0 2 3 6】

具体的には、ピクチャヘッダにおいて、第3の構文要素(ここではpic_ccalf_enabled_flagで示す)が、第2の構文要素の値に応じて提供され得る。この構文要素は、(以下で説明するように)CCALFが現在のピクチャに対して有効にされるべきかどうかを示し得る。

【0237】

第4の構文要素(pic_cross_component_alf_cb_enabled_flagなど)が、第2の構文要素に応じてピクチャヘッダにおいてさらに提供され得る。

【0238】

たとえば、第2の構文要素の値ならびに/または第3の構文要素の値および/もしくは第4の構文要素の値に応じて、pic_cross_component_alf_cb_aps_idのような第5の構文要素が提供され得る。さらに、ここではpic_cross_component_cb_filters_signalled_minus1で示す第6の構文要素が、ピクチャヘッダにおいて、潜在的には第2の構文要素および/または第3の構文要素および/または第4の構文要素にも応じて提供され得る。

【0239】

これと並行して、上記でpic_cross_component_alf_cr_enabled_flagで示した第7の構文要素が、第2の構文要素および/または第3の構文要素に応じて提供され得る。

【0240】

たとえば、第7の構文要素の値に応じて、しかし、潜在的には第2の構文要素および/または第3の構文要素にも応じて、第8の構文要素(pic_cross_component_alf_cr_aps_idなど)および/または第9の構文要素(pic_cross_component_cr_filters_signalled_minus1)が提供され得る。

【0241】

第1から第9の構文要素の意味、および以下でさらに指定する第10から第14の構文要素の意味についても、後で説明する。

【0242】

スライスヘッダ構文は、以下のとおりである。

【0243】

10

20

30

40

50

【表 6】

7.3.7.1 一般的なスライスヘッダ構文

slice_header() {	記述子
.....	
if(sps_alf_enabled_flag && !pic_alf_enabled_present_flag) {	
slice_alf_enabled_flag	u(1)
if(slice_alf_enabled_flag) {	
slice_num_alf_aps_ids_luma	u(3)
for(i = 0; i < slice_num_alf_aps_ids_luma; i++)	
slice_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
slice_alf_chroma_idc	u(2)
if(slice_alf_chroma_idc)	
slice_alf_aps_id_chroma	u(3)
}	
if(sps_ccalf_enabled_flag && ! pic_ccalf_enabled_present_flag) {	
slice_ccalf_enabled_flag	u(1)
if(slice_ccalf_enabled_flag) {	
if(ChromaArrayType != 0)	
slice_cross_component_alf_cb_enabled_flag	u(1)
if(slice_cross_component_alf_cb_enabled_flag) {	
slice_cross_component_alf_cb_aps_id	u(3)
slice_cross_component_cb_filters_signalled_minus1	ue(v)
}	
if(ChromaArrayType != 0)	
slice_cross_component_alf_cr_enabled_flag	u(1)
if(slice_cross_component_alf_cr_enabled_flag) {	
slice_cross_component_alf_cr_aps_id	u(3)
slice_cross_component_cr_filters_signalled_minus1	ue(v)
}	
}	
}	
.....	

【0 2 4 4】

上記で提供したように、いくつかの実施形態において、スライスヘッダ構文において提供されたさらなる構文要素は、ピクチャヘッダにおいて提供された第1および/もしくは第2の構文要素ならびに/または1つもしくは複数のさらなる構文要素が適切な値を取る場合にのみ提供され得る。

【0 2 4 5】

具体的には、上記で示したように、slice_cross_component_alf_cb_enabled_flagのような第10の構文要素が、第2の構文要素の値に応じて、かつ0に等しくない(SPSのレベルにおいてオプションで提供され、ここではChromaArrayTypeで示される)第14の構文要素にも潜在的に応じて提供され得る。さらに、上記でslice_cross_component_alf_cb_aps_idで示す第11の構文要素が、第2の構文要素の値に応じて、および/または第10の構文要素の値に応じて提供され得る。

【0 2 4 6】

これに対応して、slice_cross_component_alf_cr_enabled_flagのような第12の構

文要素が、第2の構文要素の値に応じて、かつ0に等しくない第14の構文要素にも潜在的に応じて提供され得る。さらに、上記でslice_cross_component_alf_cr_aps_idで示す第13の構文要素が、第2の構文要素の値に応じて、および/または第12の構文要素の値に応じて提供され得る。

【0247】

しかしながら、スライスヘッダにおいて提供される構文要素(特に、第10から第13の構文要素)は、第2の構文要素または第14の構文要素などの他の構成要素の値から独立して存在し得ることも包含される。具体的には、スライスヘッダ内のCC-ALFに属する(たとえば、第10から第13の構文要素のような)構文要素は、第1の構文要素または第2の構文要素または別の構文要素が、CC-ALFが有効にされていないことを示す場合、デフォルト値を取ることが規定され得る。

10

【0248】

新しく導入された構文要素のセマンティクスは、以下のとおりである。

【0249】

第2の構文要素は、ここではsps_ccalf_enabled_flagで示される。これは、単なる例示であり、本開示を制限していない。実施形態において、0に等しい第2の構文要素の値は、クロスコンポーネント適応ループフィルタが現在のビデオシーケンスに対して無効にされていることを指定する。1に等しい第2の構文要素の値は、クロスコンポーネント適応ループフィルタが現在のビデオシーケンスに対して有効にされていることを指定する。

【0250】

1に等しいpic_ccalf_enabled_present_flagは、pic_ccalf_enabled_flag、pic_cross_component_alf_cb_enabled_flag、pic_cross_component_alf_cb_aps_id、pic_cross_component_alf_cb_filter_count_minus1、pic_cross_component_alf_cr_enabled_flag、pic_cross_component_alf_cr_aps_id、およびpic_cross_component_alf_cr_filter_count_minus1がPH(ピクチャヘッダ)内に存在することを指定する。0に等しいpic_alf_enabled_present_flagは、pic_ccalf_enabled_flag、pic_cross_component_alf_cb_enabled_flag、pic_cross_component_alf_cb_aps_id、pic_cross_component_alf_cb_filter_count_minus1、pic_cross_component_alf_cr_enabled_flag、pic_cross_component_alf_cr_aps_id、およびpic_cross_component_alf_cr_filter_count_minus1がPH内に存在しないことを指定する。pic_ccalf_enabled_present_flagが存在しない場合、0に等しいと推定される。

20

30

【0251】

1に等しい(ここではpic_ccalf_enabled_flagで示す)第3の構文要素の値は、クロスコンポーネント適応ループフィルタが、PHに関連付けられたすべてのスライスに対して有効にされており、スライス内のCbまたはCr色成分に適用され得ることを指定する。0に等しい第3の構文要素の値は、クロスコンポーネント適応ループフィルタが、PH(ピクチャヘッダ)に関連付けられた1つ、または複数、またはすべてのスライスに対して無効にされ得ることを指定する。存在しない場合、第3の構文要素の値は、0に等しいと推定され得る。

【0252】

第4の構文要素は、上記で言及されており、ここでは、pic_cross_component_alf_cb_enabled_flagで示され得る。0に等しい第4の構文要素の値は、クロスコンポーネントCbフィルタが、PHに関連付けられたすべてのスライスのCb色成分に適用されないことを指定する。第4の構文要素の値が1に等しい場合、これは、クロスコンポーネントCbフィルタが、PHに関連付けられたすべてのスライスのCb色成分に適用されることを示す。第4の構文要素が存在しない場合、0に等しいと推定される。

40

【0253】

第5の構文要素は、ここではpic_cross_component_alf_cb_aps_idで示され得る。第5の構文要素は、PHに関連付けられたすべてのスライスのCb色成分にadaptation_parameter_set_idを指定する。

【0254】

50

第6の構文要素は、ここではpic_cross_component_cb_filters_signalled_minus1で示され得る。第6の構文要素に1を加えた値は、PHに関連付けられたすべてのスライスのクロスコンポーネントCbフィルタの数を指定する。第6の構文要素の値は、0から3の範囲内であるべきである。

【0255】

第4の構文要素が1に等しい場合、第6の構文要素が現在のピクチャの第5の構文要素によって参照されるALF APS内のalf_cross_component_cb_filters_signalled_minus1の値以下であるべきであることが、ビットストリーム適合の要件である。

【0256】

上記で言及した第7の構文要素は、pic_cross_component_alf_cr_enabled_flagで示される。0に等しい第7の構文要素の値は、クロスコンポーネントCrフィルタがPHに関連付けられたすべてのスライスのCr色成分に適用されないことを指定する。1に等しい第7の構文要素の値は、クロスコンポーネントCrフィルタがPHに関連付けられたすべてのスライスのCr色成分に適用されることを指定する。第7の構文要素が存在しない場合、0に等しいと推定される。

【0257】

第8の構文要素も言及され、pic_cross_component_alf_cr_aps_idで例示的に示された。第8の構文要素の値は、PHに関連付けられたすべてのスライスのCr色成分にadaptation_parameter_set_idを指定することを意図している。

【0258】

さらに、pic_cross_component_cr_filters_signalled_minus1で示され得る第9の構文要素について言及する。第9の構文要素に1を加えた値は、PHに関連付けられたすべてのスライスのクロスコンポーネントCrフィルタの数を指定する。第9の構文要素の値は、0から3の範囲内であるべきである。

【0259】

第7の構文要素の値が1に等しい場合、第9の構文要素の値が現在のピクチャの第8の構文要素によって参照されるALF APS内のalf_cross_component_cr_filters_signalled_minus1の値以下であるべきであることが、ビットストリーム適合の要件である。

【0260】

CCALFスライスヘッダ構文要素のセマンティクスは、以下のようにさらに調整される。

【0261】

1に等しいslice_ccalf_enabled_flagは、クロスコンポーネント適応ループフィルタが有効にされており、スライス内のCbまたはCr色成分に適用され得ることを指定する。0に等しいslice_alf_enabled_flagは、クロスコンポーネント適応ループフィルタがスライス内のすべての色成分に対して無効にされていることを指定する。存在しない場合、slice_ccalf_enabled_flagの値は、第3の構文要素と等しいと推定される。

【0262】

上記で言及した第10の構文要素は、slice_cross_component_alf_cb_enabled_flagで示され得、0に等しい第10の構文要素の値が、クロスコンポーネントCbフィルタがCb色成分に適用されないことを指定することが規定され得る。1に等しい第10の構文要素は、クロスコンポーネントCbフィルタがCb色成分に適用されることを示す。第10の構文要素が存在しない場合、第4の構文要素(pic_cross_component_alf_cb_enabled_flagなど)に等しいと想定される。

【0263】

上記で言及した第11の構文要素は、slice_cross_component_alf_cb_aps_idと呼ばれ得る。第11の構文要素の値は、スライスのCb色成分が参照するadaptation_parameter_set_idを指定する。

【0264】

第10の構文要素が1に等しい値を有する場合、現在のピクチャのすべてのスライスについて、第11の構文要素によって参照されるALF APSが同じであるべきであることが、ビッ

10

20

30

40

50

トストリーム適合の要件である。

【0265】

第10の構文要素が1に等しく、第11の構文要素が存在しない場合、第11の構文要素の値は、第5の構文要素の値に等しいと推定される。

【0266】

slice_cross_component_cb_filters_signalled_minus1に1を加えたものは、クロスコンポーネントCbフィルタの数を指定する。slice_cross_component_cb_filters_signalled_minus1の値は、0から3の範囲内であるべきである。

【0267】

第10の構文要素が1に等しい場合、slice_cross_component_cb_filters_signalled_minus1が現在のスライスの第11のものによって参照されるALF APS内のalf_cross_component_cb_filters_signalled_minus1の値以下であるべきであることが、ビットストリーム適合の要件である。

【0268】

第10の構文要素が1に等しく、第11のものが存在しない場合、slice_cross_component_cb_filters_signalled_minus1の値は、pic_cross_component_cb_filters_signalled_minus1の値に等しいと推定される。

【0269】

言及した第12の構文要素は、slice_cross_component_alf_cr_enabled_flagと呼ばれ得る。0に等しい第12の構文要素の値は、クロスコンポーネントCrフィルタがCr色成分に適用されないことを指定する。1に等しい第12の構文要素の値は、クロスコンポーネント適応ループフィルタがCr色成分に適用されることを示す。第12の構文要素が存在しない場合、pic_cross_component_alf_cr_enabled_flagのような第7の構文要素に等しいと推定される。

【0270】

第13の構文要素が、言及されており、さらにslice_cross_component_alf_cr_aps_idと呼ばれ得る。第13の構文要素の値は、スライスのCr色成分が参照するadaptation_parameter_set_idを指定する。

【0271】

第12の構文要素の値が1に等しい場合、現在のピクチャのすべてのスライスについて、第13の構文要素によって参照されるALF APSが同じであるべきであることが、ビットストリーム適合の要件である。

【0272】

第12の構文要素の値が1に等しく、第13の構文要素が存在しない場合、第13の構文要素の値は、第8の構文要素の値に等しいと推定される。

【0273】

slice_cross_component_cr_filters_signalled_minus1に1を加えたものは、クロスコンポーネントCrフィルタの数を指定する。slice_cross_component_cr_filters_signalled_minus1の値は、0から3の範囲内であるべきである。

【0274】

第12の構文要素が1に等しい場合、slice_cross_component_cr_filters_signalled_minus1が現在のスライスの第13の構文要素によって参照される参照ALF APS内のalf_cross_component_cr_filters_signalled_minus1の値以下であるべきであることが、ビットストリーム適合の要件である。

【0275】

第12の構文要素が1に等しく、第13の構文要素が存在しない場合、slice_cross_component_cr_filters_signalled_minus1の値は、pic_cross_component_cr_filters_signalled_minus1の値に等しいと推定される。

【0276】

実施形態代替案2

10

20

30

40

50

代替案2において、SPSレベルにおける第2の構文要素(sps_ccalf_enabled_flagなど)は、もう導入されず、したがって、第1の構文要素(sps_alf_enabled_flag)も、ccalfフィルタの適用を制御する。

【0277】

代替案2に関する構文は、以下のとおりである。

【0278】

【表7】

7.3.2.6 ピクチャヘッダ RBSP 構文

picture_header_rbsp() {	記述子	
.....		
if(sps_alf_enabled_flag) {		
pic_alf_enabled_present_flag	u(1)	
if(pic_alf_enabled_present_flag) {		
pic_alf_enabled_flag	u(1)	
if(pic_alf_enabled_flag) {		
pic_num_alf_aps_ids_luma	u(3)	
for(i = 0; i < pic_num_alf_aps_ids_luma; i++)		
pic_alf_aps_id_luma[i]	u(3)	
if(ChromaArrayType != 0)		
pic_alf_chroma_idc	u(2)	
if(pic_alf_chroma_idc)		
pic_alf_aps_id_chroma	u(3)	
}		
}		
pic_ccalf_enabled_present_flag	u(1)	
if(pic_ccalf_enabled_present_flag) {		
pic_ccalf_enabled_flag	u(1)	
if(pic_ccalf_enabled_flag) {		
if(ChromaArrayType != 0)		
pic_cross_component_alf_cb_enabled_flag	u(1)	
if(pic_cross_component_alf_cb_enabled_flag) {		
pic_cross_component_alf_cb_aps_id	u(3)	
}	ue(v)	
pic_cross_component_cb_filters_signalled_minus1		
}		
if(ChromaArrayType != 0)		
pic_cross_component_alf_cr_enabled_flag	u(1)	
if(pic_cross_component_alf_cr_enabled_flag) {		
pic_cross_component_alf_cr_aps_id	u(3)	
}	ue(v)	
pic_cross_component_cr_filters_signalled_minus1		
}		
}		
}		
.....		

【0279】

この第2の実施形態において、sps_ccalf_enabled_flagであり得る第2の構文要素は、SPSレベルにおいて提供され得る。CCALFが有効にされるべきかどうかは、この実施形態において、CCALFに属するさらなる構文要素が存在するかどうかを指定し、そこから、pi

c_ccalf_enabled_present_flagが属する現在のピクチャについて、CCALFが有効にされるべきであることがさらに得られ得る、pic_ccalf_enabled_present_flagのようなさらなる構文要素に基づいて指定され得る。

【0280】

スライスヘッダ構文は、以下のとおりである。

【0281】

【表8】

7.3.7.1 一般的なスライスヘッダ構文

slice_header() {	記述子
.....	
if(sps_alf_enabled_flag && !pic_alf_enabled_present_flag) {	
slice_alf_enabled_flag	u(1)
if(slice_alf_enabled_flag) {	
slice_num_alf_aps_ids_luma	u(3)
for(i = 0; i < slice_num_alf_aps_ids_luma; i++)	
slice_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
slice_alf_chroma_idc	u(2)
if(slice_alf_chroma_idc)	
slice_alf_aps_id_chroma	u(3)
}	
if(sps_alf_enabled_flag && !pic_ccalf_enabled_present_flag)	
{	
slice_ccalf_enabled_flag	u(1)
if(slice_ccalf_enabled_flag) {	
if(ChromaArrayType != 0)	
slice_cross_component_alf_cb_enabled_flag	u(1)
if(slice_cross_component_alf_cb_enabled_flag) {	
slice_cross_component_alf_cb_aps_id	u(3)
slice_cross_component_cb_filters_signalled_minus1	ue(v)
}	
if(ChromaArrayType != 0)	
slice_cross_component_alf_cr_enabled_flag	u(1)
if(slice_cross_component_alf_cr_enabled_flag) {	
slice_cross_component_alf_cr_aps_id	u(3)
slice_cross_component_cr_filters_signalled_minus1	ue(v)
}	
}	
.....	

【0282】

上記からわかるように、たとえば、slice_ccalf_enabled_flag、slice_cross_component_alf_cb_enabled_flag、slice_cross_component_alf_cb_aps_id、slice_cross_component_cb_filters_signalled_minus1、slice_cross_component_alf_cr_enabled_flag、slice_cross_component_alf_cr_aps_id、およびslice_cross_component_cr_filters_signalled_minus1、ならびに/または上記ですでに言及した第10から第13の構文要素などのスライスヘッダ内の構文要素の存在は、sps_alf_enabled_flagのようなSPSレベルにおいて提供される第1の構文要素、およびpic_ccalf_enabled_present_fl

agのようなピクチャヘッダにおいて提供される構文要素、ならびに、潜在的には、slice_ccalf_enabled_flagのようなスライスヘッダにおいて提供されるさらなる構文要素に依存し得る。

【0283】

実施形態代替案3

CCALFピクチャヘッダエントリに関する他の代替案は、以下のとおりである。この代替案において、pic_alf_enabled_present_flag、pic_alf_enabled_flagは、それぞれ、CCALFピクチャヘッダおよびCCALFスライスヘッダエントリを制御する。

【0284】

【表9】

10

7.3.2.6 ピクチャヘッダ RBSP 構文

picture_header_rbsp() {	記述子
.....	
if(sps_alf_enabled_flag) {	
pic_alf_enabled_present_flag	u(1)
if(pic_alf_enabled_present_flag) {	
pic_alf_enabled_flag	u(1)
if(pic_alf_enabled_flag) {	
pic_num_alf_aps_ids_luma	u(3)
for(i = 0; i < pic_num_alf_aps_ids_luma; i++)	
pic_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
pic_alf_chroma_idc	u(2)
if(pic_alf_chroma_idc)	
pic_alf_aps_id_chroma	u(3)
if(ChromaArrayType != 0)	
pic_cross_component_alf_cb_enabled_flag	u(1)
if(pic_cross_component_alf_cb_enabled_flag) {	
pic_cross_component_alf_cb_aps_id	u(3)
pic_cross_component_cb_filters_signalled_minus1	ue(v)
}	
if(ChromaArrayType != 0)	
pic_cross_component_alf_cr_enabled_flag	u(1)
if(pic_cross_component_alf_cr_enabled_flag) {	
pic_cross_component_alf_cr_aps_id	u(3)
pic_cross_component_cr_filters_signalled_minus1	ue(v)
} /* end of pic_alf_enabled_flag */	
} /* end of pic_alf_enabled_present_flag */	
} /* end of sps_alf_enabled_flag */	
.....	

20

30

40

【0285】

スライスヘッダ構文は、以下のとおりである。

【0286】

50

【表 10】

7.3.7.1 一般的なスライスヘッダ構文

slice_header() {	記述子
.....	
if(sps_alf_enabled_flag && !pic_alf_enabled_present_flag) //explanation: if sps_alf_enabled_flag is true (and) if pic_alf_enabled_present_flag is false){	
slice_alf_enabled_flag	u(1)
if(slice_alf_enabled_flag) {	
slice_num_alf_aps_ids_luma	u(3)
for(i = 0; i < slice_num_alf_aps_ids_luma; i++)	
slice_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
slice_alf_chroma_idc	u(2)
if(slice_alf_chroma_idc)	
slice_alf_aps_id_chroma	u(3)
if(ChromaArrayType != 0)	
slice_cross_component_alf_cb_enabled_flag	u(1)
if(slice_cross_component_alf_cb_enabled_flag) {	
slice_cross_component_alf_cb_aps_id	u(3)
slice_cross_component_cb_filters_signalled_minus1	ue(v)
}	
if(ChromaArrayType != 0)	
slice_cross_component_alf_cr_enabled_flag	u(1)
if(slice_cross_component_alf_cr_enabled_flag) {	
slice_cross_component_alf_cr_aps_id	u(3)
slice_cross_component_cr_filters_signalled_minus1	ue(v)
}	
} /* end of slice_alf_enabled_flag loop */	
} /* end of sps_alf_enabled_flag && !pic_alf_enabled_present_flag */	
.....	

【0287】

!(0)=1または!(1)=0であることが留意され得る。

【0288】

実施形態代替案4

CCALFの現在の設計において、slice_cross_component_alf_cb_aps_id、slice_cross_component_alf_cr_aps_idがすべてのスライスにわたって同じであるべきであるという制限があるので、スライスヘッダにおいてこの構文要素を繰り返すことは意味がなく、ピクチャヘッダにおいてpic_cross_component_alf_cb_aps_idおよびpic_cross_component_alf_cr_aps_idをシグナリングするだけである。このとき、構文要素slice_cross_component_alf_cb_aps_idおよびslice_cross_component_alf_crの値は、pic_cross_component_alf_cb_aps_idおよびpic_cross_component_alf_cr_aps_idと同じであると推定され得る。これによって、ビットストリーム内の情報量は、さらに削減され得る。

【0289】

可能な構文は、以下のとおりである。

【0290】

【表11】

7.3.2.6 ピクチャヘッダ RBSP 構文

picture_header_rbsp() {	記述子
.....	
if(sps_alf_enabled_flag) {	
pic_alf_enabled_present_flag	u(1)
if(pic_alf_enabled_present_flag) {	
pic_alf_enabled_flag	u(1)
if(pic_alf_enabled_flag) {	
pic_num_alf_aps_ids_luma	u(3)
for(i = 0; i < pic_num_alf_aps_ids_luma; i++)	
pic_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
pic_alf_chroma_idc	u(2)
if(pic_alf_chroma_idc)	
pic_alf_aps_id_chroma	u(3)
}	
}	
if(sps_ccalf_enabled_flag) {	
pic_cross_component_alf_cb_aps_id	u(3)
pic_cross_component_alf_cr_aps_id	u(3)
pic_ccalf_enabled_present_flag	u(1)
if(pic_ccalf_enabled_present_flag) {	
pic_ccalf_enabled_flag	u(1)
if(pic_ccalf_enabled_flag) {	
if(ChromaArrayType != 0)	
pic_cross_component_alf_cb_enabled_flag	u(1)
if(pic_cross_component_alf_cb_enabled_flag) {	
pic_cross_component_cb_filters_signalled_minus1	ue(v)
}	
if(ChromaArrayType != 0)	
pic_cross_component_alf_cr_enabled_flag	u(1)
if(pic_cross_component_alf_cr_enabled_flag) {	
pic_cross_component_cr_filters_signalled_minus1	ue(v)
}	
}	
}	
.....	

【0291】

ここで、CCALFに属するさらなる構文要素(すなわち、少なくとも要素ccalfまたはcross_component_alfまたはcc_alfを含む構文要素は、SPSレベルにおいてシグナリングされる第2の構文要素、すなわち、この実施形態においてsps_ccalf_enabled_flagの値に応じて提供される。

【0292】

具体的には、第2の構文要素の値に応じて、pic_ccalf_enabled_flagのような第3の構

文要素がピクチャヘッダにおいて提供され得、この第3の構文要素は、CCALFが現在のピクチャのスライスに対して有効にされているかどうかを示し得る。

【0293】

さらに、pic_cross_component_alf_cb_enabled_flagのような第4の構文要素、およびpic_cross_component_alf_cb_enabled_flagのような第7の構文要素が、第2の構文要素の値に応じてピクチャヘッダにおいて提供され得る。

【0294】

スライスヘッダ構文は、以下のとおりであり得る。

【0295】

【表12】

10

7.3.7.1 一般的なスライスヘッダ構文

slice_header() {	記述子
.....	
if(sps_alf_enabled_flag && !pic_alf_enabled_present_flag) {	
slice_alf_enabled_flag	u(1)
if(slice_alf_enabled_flag) {	
slice_num_alf_aps_ids_luma	u(3)
for(i = 0; i < slice_num_alf_aps_ids_luma; i++)	
slice_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
slice_alf_chroma_idc	u(2)
if(slice_alf_chroma_idc)	
slice_alf_aps_id_chroma	u(3)
}	
if(sps_ccalf_enabled_flag && !pic_ccalf_enabled_present_flag) {	
slice_ccalf_enabled_flag	u(1)
if(slice_ccalf_enabled_flag) {	
if(ChromaArrayType != 0)	
slice_cross_component_alf_cb_enabled_flag	u(1)
if(slice_cross_component_alf_cb_enabled_flag) {	
slice_cross_component_cb_filters_signalled_minus1	ue(v)
}	
if(ChromaArrayType != 0)	
slice_cross_component_alf_cr_enabled_flag	u(1)
if(slice_cross_component_alf_cr_enabled_flag) {	
slice_cross_component_cr_filters_signalled_minus1	ue(v)
}	
}	
.....	

20

30

40

【0296】

また、この実施形態について、CCALFに属するスライスヘッダ内の構文要素が、SPSレベルにおいて提供される第2の構文要素の値に応じて、および/または、たとえば、pic_ccalf_enabled_present_flagのような、ピクチャヘッダ内で提供されるCCALFに属する1つもしくは複数の構文要素の値に応じて提供されることが規定され得る。

【0297】

slice_cross_component_alf_cb_aps_idは、常にpic_cross_component_alf_cb_a

50

ps_idの値と同じであると推定され得る。

【0298】

slice_cross_component_alf_cr_aps_idは、常にpic_cross_component_alf_cr_aps_idの値と同じであると推定され得る。

【0299】

他の可能な構文は、以下のとおりである。

【0300】

【表13】

7.3.2.6 ピクチャヘッダ RBSP 構文

picture_header_rbsp() {	記述子
.....	
if(sps_alf_enabled_flag) {	
pic_alf_enabled_present_flag	u(1)
if(pic_alf_enabled_present_flag) {	
pic_alf_enabled_flag	u(1)
if(pic_alf_enabled_flag) {	
pic_num_alf_aps_ids_luma	u(3)
for(i = 0; i < pic_num_alf_aps_ids_luma; i++)	
pic_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
pic_alf_chroma_idc	u(2)
if(pic_alf_chroma_idc)	
pic_alf_aps_id_chroma	u(3)
}	
}	
pic_cross_component_alf_cb_aps_id	u(3)
pic_cross_component_alf_cr_aps_id	u(3)
pic_ccalf_enabled_present_flag	u(1)
if(pic_ccalf_enabled_present_flag) {	
pic_ccalf_enabled_flag	u(1)
if(pic_ccalf_enabled_flag) {	
if(ChromaArrayType != 0)	
pic_cross_component_alf_cb_enabled_flag	u(1)
if(pic_cross_component_alf_cb_enabled_flag) {	
pic_cross_component_alf_cb_aps_id	u(3)
}	ue(v)
pic_cross_component_cb_filters_signalled_minus1	
}	
if(ChromaArrayType != 0)	
pic_cross_component_alf_cr_enabled_flag	u(1)
if(pic_cross_component_alf_cr_enabled_flag) {	
pic_cross_component_alf_cr_aps_id	u(3)
}	ue(v)
pic_cross_component_cr_filters_signalled_minus1	
}	
}	
}	
.....	

【0301】

上記の表による構文において、CCALFに属する構文要素の存在は、sps_alf_enabled_f

lagのようなSPSレベルにおいて提供される第1の構文要素の値に依存する。sps_ccalf_enabled_flagのような第2の構文要素は、この実施形態において、SPSレベルにおいて提供されてもされなくてもよい。

【0302】

スライスヘッダ構文は、以下のとおりであり得る。

【0303】

【表14】

7.3.7.1 一般的なスライスヘッダ構文

slice_header() {	記述子	
.....		
if(sps_alf_enabled_flag && !pic_alf_enabled_present_flag) {		
slice_alf_enabled_flag	u(1)	
if(slice_alf_enabled_flag) {		
slice_num_alf_aps_ids_luma	u(3)	
for(i = 0; i < slice_num_alf_aps_ids_luma; i++)		
slice_alf_aps_id_luma[i]	u(3)	
if(ChromaArrayType != 0)		
slice_alf_chroma_idc	u(2)	
if(slice_alf_chroma_idc)		
slice_alf_aps_id_chroma	u(3)	
}		
if(sps_alf_enabled_flag && !pic_ccalf_enabled_present_flag)		
{		
slice_ccalf_enabled_flag	u(1)	
if(slice_ccalf_enabled_flag) {		
if(ChromaArrayType != 0)		
slice_cross_component_alf_cb_enabled_flag	u(1)	
if(slice_cross_component_alf_cb_enabled_flag) {		
slice_cross_component_cb_filters_signalled_minus1	ue(v)	
}		
if(ChromaArrayType != 0)		
slice_cross_component_alf_cr_enabled_flag	u(1)	
if(slice_cross_component_alf_cr_enabled_flag) {		
slice_cross_component_cr_filters_signalled_minus1	ue(v)	
}		
}		
.....		

【0304】

slice_cross_component_alf_cb_aps_idは、常にpic_cross_component_alf_cb_aps_idの値と同じであると推定される。

【0305】

slice_cross_component_alf_cr_aps_idは、常にpic_cross_component_alf_cr_aps_idの値と同じであると推定される。

【0306】

他の可能な構文は、以下のとおりである。

【0307】

10

20

30

40

50

【表 15】

ピクチャヘッダ RBSP 構文

picture_header_rbsp() {	記述子
.....	
if(sps_alf_enabled_flag) {	
<i>pic_cross_component_alf_cb_aps_id</i>	u(3)
<i>pic_cross_component_alf_cr_aps_id</i>	u(3)
<i>pic_alf_enabled_present_flag</i>	u(1)
if(pic_alf_enabled_present_flag) {	
<i>pic_alf_enabled_flag</i>	u(1)
if(pic_alf_enabled_flag) {	
<i>pic_num_alf_aps_ids_luma</i>	u(3)
for(i = 0; i < pic_num_alf_aps_ids_luma; i++)	
<i>pic_alf_aps_id_luma[i]</i>	u(3)
if(ChromaArrayType != 0)	
<i>pic_alf_chroma_idc</i>	u(2)
if(pic_alf_chroma_idc)	
<i>pic_alf_aps_id_chroma</i>	u(3)
<i>if(ChromaArrayType != 0)</i>	
<i>pic_cross_component_alf_cb_enabled_flag</i>	u(1)
<i>if(pic_cross_component_alf_cb_enabled_flag) {</i>	
<i>pic_cross_component_cb_filters_signalled_minus1</i>	ue(v)
<i>}</i>	
<i>if(ChromaArrayType != 0)</i>	
<i>pic_cross_component_alf_cr_enabled_flag</i>	u(1)
<i>if(pic_cross_component_alf_cr_enabled_flag) {</i>	
<i>pic_cross_component_cr_filters_signalled_minus1</i>	ue(v)
<i>}</i> /* end of pic_alf_enabled_flag */	
<i>}</i> /* end of pic_alf_enabled_present_flag */	
<i>}</i> /* end of sps_alf_enabled_flag */	
.....	

【0308】

スライスヘッダ構文は、以下のとおりである。

【0309】

10

20

30

40

50

【表 16】

7.3.7.1 一般的なスライスヘッダ構文

slice_header() {	記述子
.....	
if(sps_alf_enabled_flag && !pic_alf_enabled_present_flag) {	
slice_alf_enabled_flag	u(1)
if(slice_alf_enabled_flag) {	
slice_num_alf_aps_ids_luma	u(3)
for(i = 0; i < slice_num_alf_aps_ids_luma; i++)	
slice_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
slice_alf_chroma_idc	u(2)
if(slice_alf_chroma_idc)	
slice_alf_aps_id_chroma	u(3)
if(ChromaArrayType != 0)	
slice_cross_component_alf_cb_enabled_flag	u(1)
if(slice_cross_component_alf_cb_enabled_flag) {	
slice_cross_component_cb_filters_signalled_minus1	ue(v)
}	
if(ChromaArrayType != 0)	
slice_cross_component_alf_cr_enabled_flag	u(1)
if(slice_cross_component_alf_cr_enabled_flag) {	
slice_cross_component_cr_filters_signalled_minus1	ue(v)
}	
} /* end of slice_alf_enabled_flag loop */	
} /* end of sps_alf_enabled_flag && !pic_alf_enabled_present_flag */	
.....	

【0310】

この実施形態において、slice_cross_component_alf_cb_aps_idは、常にpic_cross_component_alf_cb_aps_idの値と同じであると推定される。

【0311】

この実施形態において、slice_cross_component_alf_cr_aps_idのような第12の構文要素は、常にpic_cross_component_alf_cr_aps_id)のような第8の構文要素の値と同じであると推定される。

【0312】

実施形態代替案5

他の代替構文は、可能な限り以下のとおりである。この構文において、CCALFパラメータは、上記ですでに言及したように、第2の構文要素の値に基づいて条件付きでシグナリングされる。この第2の構文要素は、sps_ccalf_enabled_flagなどのフラグとして提供され得るか、またはフラグを含み得る。

【0313】

10

20

30

40

50

【表 1 7 A】

シーケンスパラメータセット RBSP 構文

seq_parameter_set_rbsp() {	記述子
sps_decoding_parameter_set_id	u(4)
sps_video_parameter_set_id	u(4)
sps_max_sublayers_minus1	u(3)
sps_reserved_zero_4bits	u(4)
sps_ptl_dpb_hrd_params_present_flag	u(1)
if(sps_ptl_dpb_hrd_params_present_flag)	
profile_tier_level(1, sps_max_sublayers_minus1)	
gdr_enabled_flag	u(1)
sps_seq_parameter_set_id	u(4)
chroma_format_idc	u(2)
if(chroma_format_idc == 3)	
separate_colour_plane_flag	u(1)
ref_pic_resampling_enabled_flag	u(1)
pic_width_max_in_luma_samples	ue(v)
pic_height_max_in_luma_samples	ue(v)
sps_log2_ctu_size_minus5	u(2)
subpics_present_flag	u(1)
if(subpics_present_flag) {	
sps_num_subpics_minus1	u(8)
for(i = 0; i <= sps_num_subpics_minus1; i++) {	
subpic_ctu_top_left_x[i]	u(v)
subpic_ctu_top_left_y[i]	u(v)
subpic_width_minus1[i]	u(v)
subpic_height_minus1[i]	u(v)
subpic_treated_as_pic_flag[i]	u(1)
loop_filter_across_subpic_enabled_flag[i]	u(1)
}	
}	
sps_subpic_id_present_flag	u(1)
if(sps_subpics_id_present_flag) {	
sps_subpic_id_signalling_present_flag	u(1)
if(sps_subpics_id_signalling_present_flag) {	
sps_subpic_id_len_minus1	ue(v)
for(i = 0; i <= sps_num_subpics_minus1; i++)	
sps_subpic_id[i]	u(v)
}	
}	
bit_depth_minus8	ue(v)
min_qp_prime_ts_minus4	ue(v)
sps_weighted_pred_flag	u(1)

10

20

30

40

50

【表 1 7 B】

sps_weighted_bipred_flag	u(1)	
log2_max_pic_order_cnt_lsb_minus4	u(4)	
sps_poc_msb_flag	u(1)	
if(sps_poc_msb_flag)		
poc_msb_len_minus1	ue(v)	
if(sps_max_sublayers_minus1 > 0)		
sps_sublayer_dpb_params_flag	u(1)	
if(sps_ptl_dpb_hrd_params_present_flag)		
dpb_parameters(0, sps_max_sublayers_minus1, sps_sublayer_dpb_params_flag)		
long_term_ref_pics_flag	u(1)	
inter_layer_ref_pics_present_flag	u(1)	
sps_idr_rpl_present_flag	u(1)	
rpl1_same_as_rpl0_flag	u(1)	
for(i = 0; i < !rpl1_same_as_rpl0_flag ? 2 : 1; i++) {		
num_ref_pic_lists_in_sps[i]	ue(v)	
for(j = 0; j < num_ref_pic_lists_in_sps[i]; j++)		
ref_pic_list_struct(i, j)		
}		
if(ChromaArrayType != 0)		
qtbtt_dual_tree_intra_flag	u(1)	
log2_min_luma_coding_block_size_minus2	ue(v)	
partition_constraints_override_enabled_flag	u(1)	
sps_log2_diff_min_qt_min_cb_intra_slice_luma	ue(v)	
sps_log2_diff_min_qt_min_cb_inter_slice	ue(v)	
sps_max_mtt_hierarchy_depth_inter_slice	ue(v)	
sps_max_mtt_hierarchy_depth_intra_slice_luma	ue(v)	
if(sps_max_mtt_hierarchy_depth_intra_slice_luma != 0) {		
sps_log2_diff_max_bt_min_qt_intra_slice_luma	ue(v)	
sps_log2_diff_max_tt_min_qt_intra_slice_luma	ue(v)	
}		
if(sps_max_mtt_hierarchy_depth_inter_slice != 0) {		
sps_log2_diff_max_bt_min_qt_inter_slice	ue(v)	
sps_log2_diff_max_tt_min_qt_inter_slice	ue(v)	
}		
if(qtbtt_dual_tree_intra_flag) {		
sps_log2_diff_min_qt_min_cb_intra_slice_chroma	ue(v)	
sps_max_mtt_hierarchy_depth_intra_slice_chroma	ue(v)	
if(sps_max_mtt_hierarchy_depth_intra_slice_chroma != 0) {		
sps_log2_diff_max_bt_min_qt_intra_slice_chroma	ue(v)	
sps_log2_diff_max_tt_min_qt_intra_slice_chroma	ue(v)	
}		
}		

10

20

30

40

50

【表 1 7 C】

sps_max_luma_transform_size_64_flag	u(1)
sps_joint_cbc_r_enabled_flag	u(1)
if(ChromaArrayType != 0) {	
same_qp_table_for_chroma	u(1)
numQpTables = same_qp_table_for_chroma ? 1 :	
(sps_joint_cbc_r_enabled_flag ? 3 : 2)	
for(i = 0; i < numQpTables; i++) {	
qp_table_start_minus26[i]	se(v)
num_points_in_qp_table_minus1[i]	ue(v)
for(j = 0; j <= num_points_in_qp_table_minus1[i]; j++) {	
delta_qp_in_val_minus1[i][j]	ue(v)
delta_qp_diff_val[i][j]	ue(v)
}	
}	
}	
sps_sao_enabled_flag	u(1)
sps_alf_enabled_flag	u(1)
if(sps_alf_enabled_flag && ChromaArrayType != 0)	
sps_ccalf_enabled_flag	u(1)
sps_transform_skip_enabled_flag	u(1)
if(sps_transform_skip_enabled_flag)	
sps_bdpcm_enabled_flag	u(1)
if(sps_bdpcm_enabled_flag && chroma_format_idc == 3)	
sps_bdpcm_chroma_enabled_flag	u(1)
sps_ref_wraparound_enabled_flag	u(1)
if(sps_ref_wraparound_enabled_flag)	
sps_ref_wraparound_offset_minus1	ue(v)
sps_temporal_mvp_enabled_flag	u(1)
if(sps_temporal_mvp_enabled_flag)	
sps_sbtmvp_enabled_flag	u(1)
sps_amvr_enabled_flag	u(1)
sps_bdof_enabled_flag	u(1)
if(sps_bdof_enabled_flag)	
sps_bdof_pic_present_flag	u(1)
sps_smvd_enabled_flag	u(1)
sps_dmvr_enabled_flag	u(1)
if(sps_dmvr_enabled_flag)	
sps_dmvr_pic_present_flag	u(1)
sps_mmvd_enabled_flag	u(1)
sps_isp_enabled_flag	u(1)
sps_mrl_enabled_flag	u(1)
sps_mip_enabled_flag	u(1)

10

20

30

40

50

【表 1 7 D】

if(ChromaArrayType != 0)	
sps_cclm_enabled_flag	u(1)
if(chroma_format_idc == 1) {	
sps_chroma_horizontal_collocated_flag	u(1)
sps_chroma_vertical_collocated_flag	u(1)
}	
sps_mts_enabled_flag	u(1)
if(sps_mts_enabled_flag) {	
sps_explicit_mts_intra_enabled_flag	u(1)
sps_explicit_mts_inter_enabled_flag	u(1)
}	
sps_sbt_enabled_flag	u(1)
sps_affine_enabled_flag	u(1)
if(sps_affine_enabled_flag) {	
sps_affine_type_flag	u(1)
sps_affine_amvr_enabled_flag	u(1)
sps_affine_prof_enabled_flag	u(1)
if(sps_affine_prof_enabled_flag)	
sps_prof_pic_present_flag	u(1)
}	
if(chroma_format_idc == 3) {	
sps_palette_enabled_flag	u(1)
sps_act_enabled_flag	u(1)
}	
sps_bcw_enabled_flag	u(1)
sps_ibc_enabled_flag	u(1)
sps_ciip_enabled_flag	u(1)
if(sps_mmvd_enabled_flag)	
sps_fpel_mmvd_enabled_flag	u(1)
sps_triangle_enabled_flag	u(1)
sps_lmcs_enabled_flag	u(1)
sps_lfnst_enabled_flag	u(1)
sps_ladf_enabled_flag	u(1)
if(sps_ladf_enabled_flag) {	
sps_num_ladf_intervals_minus2	u(2)
sps_ladf_lowest_interval_qp_offset	se(v)
for(i = 0; i < sps_num_ladf_intervals_minus2 + 1; i++) {	
sps_ladf_qp_offset[i]	se(v)
sps_ladf_delta_threshold_minus1[i]	ue(v)
}	
}	

10

20

30

40

50

【表 1 7 E】

sps_scaling_list_enabled_flag	u(1)	
sps_loop_filter_across_virtual_boundaries_disabled_present_flag	u(1)	
if(sps_loop_filter_across_virtual_boundaries_disabled_present_flag)		
{		
sps_num_ver_virtual_boundaries	u(2)	
for(i = 0; i < sps_num_ver_virtual_boundaries; i++)		
sps_virtual_boundaries_pos_x[i]	u(13)	
sps_num_hor_virtual_boundaries	u(2)	10
for(i = 0; i < sps_num_hor_virtual_boundaries; i++)		
sps_virtual_boundaries_pos_y[i]	u(13)	
}		
if(sps_ptl_dpb_hrd_params_present_flag) {		
sps_general_hrd_params_present_flag	u(1)	
if(sps_general_hrd_params_present_flag) {		
general_hrd_parameters()		
if(sps_max_sublayers_minus1 > 0)		
sps_sublayer_cpb_params_present_flag	u(1)	
firstSubLayer = sps_sublayer_cpb_params_present_flag ? 0 :		20
sps_max_sublayers_minus1		
ols_hrd_parameters(firstSubLayer, sps_max_sublayers_minus1)		
}		
}		
field_seq_flag	u(1)	
vui_parameters_present_flag	u(1)	
if(vui_parameters_present_flag)		
vui_parameters() /* Specified in ITU-T H.SEI ISO/IEC 23002-7 */		
sps_extension_flag	u(1)	
if(sps_extension_flag)		
while(more_rbsp_data())		
sps_extension_data_flag	u(1)	30
rbsp_trailing_bits()		
}		

【0 3 1 4】

上記の表において見られるように、第2の構文要素sps_ccalf_enabled_flagは、第1の構文要素sps_alf_enabled_flagの値に応じて、かつオプションでChromaArrayTypeのような第14の構文要素に応じて提供される。具体的には、第2の構文要素は、いくつかの実施形態において、第14の構文要素がゼロとは異なる値を取る場合にシグナリングされ得る。

【0 3 1 5】

10

20

30

40

50

【表 1 8】

7.3.2.3 シーケンスパラメータセット RBSP 構文

seq_parameter_set_rbsp() {	記述子
.....	
sps_alf_enabled_flag	u(1)
if(sps_alf_enabled_flag && ChromaArrayType != 0)	
sps_ccalf_enabled_flag	u(1)
.....	

10

【0 3 1 6】

注:ChromaArrayType !=0、すなわち、ルマ成分およびCb、Crクロマ成分ではない

【0 3 1 7】

ChromaArrayTypeは、以下の表において指定されているように、ルマサンプリングに対するクロマサンプリングを示す。

【0 3 1 8】

【表 1 9】

20

ChromaArrayType	クロマ フォーマット
0	モノクロ
1	4:2:0
2	4:2:2
3	4:4:4

30

【0 3 1 9】

モノクロサンプリングにおいて、名目上ルマ配列と見なされる1つのサンプル配列のみが存在する。

【0 3 2 0】

4:2:0サンプリングにおいて、2つのクロマ配列の各々は、ルマ配列の半分の高さと半分の幅とを有する。

【0 3 2 1】

4:2:2サンプリングにおいて、2つのクロマ配列の各々は、ルマ配列と同じ高さで半分の幅とを有する。

40

【0 3 2 2】

4:4:4サンプリングにおいて、2つのクロマ配列の各々は、ルマ配列と同じ高さで幅とを有する。

【0 3 2 3】

上記の表において規定されているように、第2の構文要素は、第1の構文要素が「真」を示す場合、すなわち、第1の構文要素がsps_alf_enabled_flagであり、ALFが有効にされていることを示す場合、提供される。さらに、第2の構文要素は、第14の構文要素が0に等しくない値を有する場合、提供される。

【0 3 2 4】

50

【表 2 0 A】

1.1.1.1 シーケンスパラメータセット RBSP 構文

seq_parameter_set_rbsp() {	記述子
sps_decoding_parameter_set_id	u(4)
sps_video_parameter_set_id	u(4)
sps_max_sublayers_minus1	u(3)
sps_reserved_zero_4bits	u(4)
sps_ptl_dpb_hrd_params_present_flag	u(1)
if(sps_ptl_dpb_hrd_params_present_flag)	
profile_tier_level(1, sps_max_sublayers_minus1)	
gdr_enabled_flag	u(1)
sps_seq_parameter_set_id	u(4)
chroma_format_idc	u(2)
if(chroma_format_idc == 3)	
separate_colour_plane_flag	u(1)
ref_pic_resampling_enabled_flag	u(1)
pic_width_max_in_luma_samples	ue(v)
pic_height_max_in_luma_samples	ue(v)
sps_log2_ctu_size_minus5	u(2)
subpics_present_flag	u(1)
if(subpics_present_flag) {	
sps_num_subpics_minus1	u(8)
for(i = 0; i <= sps_num_subpics_minus1; i++) {	
subpic_ctu_top_left_x[i]	u(v)
subpic_ctu_top_left_y[i]	u(v)
subpic_width_minus1[i]	u(v)
subpic_height_minus1[i]	u(v)
subpic_treated_as_pic_flag[i]	u(1)
loop_filter_across_subpic_enabled_flag[i]	u(1)
}	
}	
sps_subpic_id_present_flag	u(1)
if(sps_subpics_id_present_flag) {	
sps_subpic_id_signalling_present_flag	u(1)
if(sps_subpics_id_signalling_present_flag) {	
sps_subpic_id_len_minus1	ue(v)
for(i = 0; i <= sps_num_subpics_minus1; i++)	
sps_subpic_id[i]	u(v)
}	
}	
bit_depth_minus8	ue(v)
min_qp_prime_ts_minus4	ue(v)
sps_weighted_pred_flag	u(1)

10

20

30

40

50

【表 2 0 B】

sps_weighted_bipred_flag	u(1)
log2_max_pic_order_cnt_lsb_minus4	u(4)
sps_poc_msb_flag	u(1)
if(sps_poc_msb_flag)	
poc_msb_len_minus1	ue(v)
if(sps_max_sublayers_minus1 > 0)	
sps_sublayer_dpb_params_flag	u(1)
if(sps_ptl_dpb_hrd_params_present_flag)	
dpb_parameters(0, sps_max_sublayers_minus1, sps_sublayer_dpb_params_flag)	
long_term_ref_pics_flag	u(1)
inter_layer_ref_pics_present_flag	u(1)
sps_idr_rpl_present_flag	u(1)
rpl1_same_as_rpl0_flag	u(1)
for(i = 0; i < !rpl1_same_as_rpl0_flag ? 2 : 1; i++) {	
num_ref_pic_lists_in_sps[i]	ue(v)
for(j = 0; j < num_ref_pic_lists_in_sps[i]; j++)	
ref_pic_list_struct(i, j)	
}	
if(ChromaArrayType != 0)	
qtbtt_dual_tree_intra_flag	u(1)
log2_min_luma_coding_block_size_minus2	ue(v)
partition_constraints_override_enabled_flag	u(1)
sps_log2_diff_min_qt_min_cb_intra_slice_luma	ue(v)
sps_log2_diff_min_qt_min_cb_inter_slice	ue(v)
sps_max_mtt_hierarchy_depth_inter_slice	ue(v)
sps_max_mtt_hierarchy_depth_intra_slice_luma	ue(v)
if(sps_max_mtt_hierarchy_depth_intra_slice_luma != 0) {	
sps_log2_diff_max_bt_min_qt_intra_slice_luma	ue(v)
sps_log2_diff_max_tt_min_qt_intra_slice_luma	ue(v)
}	
if(sps_max_mtt_hierarchy_depth_inter_slice != 0) {	
sps_log2_diff_max_bt_min_qt_inter_slice	ue(v)
sps_log2_diff_max_tt_min_qt_inter_slice	ue(v)
}	
if(qtbtt_dual_tree_intra_flag) {	
sps_log2_diff_min_qt_min_cb_intra_slice_chroma	ue(v)
sps_max_mtt_hierarchy_depth_intra_slice_chroma	ue(v)
if(sps_max_mtt_hierarchy_depth_intra_slice_chroma != 0) {	
sps_log2_diff_max_bt_min_qt_intra_slice_chroma	ue(v)
sps_log2_diff_max_tt_min_qt_intra_slice_chroma	ue(v)
}	
}	

10

20

30

40

50

【表 2 0 C】

sps_max_luma_transform_size_64_flag	u(1)
sps_joint_cbr_enabled_flag	u(1)
if(ChromaArrayType != 0) {	
same_qp_table_for_chroma	u(1)
numQpTables = same_qp_table_for_chroma ? 1 :	
(sps_joint_cbr_enabled_flag ? 3 : 2)	
for(i = 0; i < numQpTables; i++) {	
qp_table_start_minus26[i]	se(v)
num_points_in_qp_table_minus1[i]	ue(v)
for(j = 0; j <= num_points_in_qp_table_minus1[i]; j++) {	
delta_qp_in_val_minus1[i][j]	ue(v)
delta_qp_diff_val[i][j]	ue(v)
}	
}	
}	
sps_sao_enabled_flag	u(1)
sps_alf_enabled_flag	u(1)
if(sps_alf_enabled_flag && ChromaArrayType != 0)	
sps_ccalf_enabled_flag	u(1)
sps_transform_skip_enabled_flag	u(1)
if(sps_transform_skip_enabled_flag)	
sps_bdpcm_enabled_flag	u(1)
if(sps_bdpcm_enabled_flag && chroma_format_idc == 3)	
sps_bdpcm_chroma_enabled_flag	u(1)
sps_ref_wraparound_enabled_flag	u(1)
if(sps_ref_wraparound_enabled_flag)	
sps_ref_wraparound_offset_minus1	ue(v)
sps_temporal_mvp_enabled_flag	u(1)
if(sps_temporal_mvp_enabled_flag)	
sps_sbtmvp_enabled_flag	u(1)
sps_amvr_enabled_flag	u(1)
sps_bdof_enabled_flag	u(1)
if(sps_bdof_enabled_flag)	
sps_bdof_pic_present_flag	u(1)
sps_smvd_enabled_flag	u(1)
sps_dmvr_enabled_flag	u(1)
if(sps_dmvr_enabled_flag)	
sps_dmvr_pic_present_flag	u(1)
sps_mmvd_enabled_flag	u(1)
sps_isp_enabled_flag	u(1)
sps_mrl_enabled_flag	u(1)
sps_mip_enabled_flag	u(1)

10

20

30

40

50

【表 2 0 D】

if(ChromaArrayType != 0)	
sps_cclm_enabled_flag	u(1)
if(chroma_format_idc == 1) {	
sps_chroma_horizontal_collocated_flag	u(1)
sps_chroma_vertical_collocated_flag	u(1)
}	
sps_mts_enabled_flag	u(1)
if(sps_mts_enabled_flag) {	
sps_explicit_mts_intra_enabled_flag	u(1)
sps_explicit_mts_inter_enabled_flag	u(1)
}	
sps_sbt_enabled_flag	u(1)
sps_affine_enabled_flag	u(1)
if(sps_affine_enabled_flag) {	
sps_affine_type_flag	u(1)
sps_affine_amvr_enabled_flag	u(1)
sps_affine_prof_enabled_flag	u(1)
if(sps_affine_prof_enabled_flag)	
sps_prof_pic_present_flag	u(1)
}	
if(chroma_format_idc == 3) {	
sps_palette_enabled_flag	u(1)
sps_act_enabled_flag	u(1)
}	
sps_bcw_enabled_flag	u(1)
sps_ibc_enabled_flag	u(1)
sps_ciip_enabled_flag	u(1)
if(sps_mmvd_enabled_flag)	
sps_fpel_mmvd_enabled_flag	u(1)
sps_triangle_enabled_flag	u(1)
sps_lmcs_enabled_flag	u(1)
sps_lfnst_enabled_flag	u(1)
sps_ladf_enabled_flag	u(1)
if(sps_ladf_enabled_flag) {	
sps_num_ladf_intervals_minus2	u(2)
sps_ladf_lowest_interval_qp_offset	se(v)
for(i = 0; i < sps_num_ladf_intervals_minus2 + 1; i++) {	
sps_ladf_qp_offset[i]	se(v)
sps_ladf_delta_threshold_minus1[i]	ue(v)
}	
}	

10

20

30

40

50

【表 2 0 E】

sps_scaling_list_enabled_flag	u(1)	
sps_loop_filter_across_virtual_boundaries_disabled_present_flag	u(1)	
if(sps_loop_filter_across_virtual_boundaries_disabled_present_flag)		
{		
sps_num_ver_virtual_boundaries	u(2)	
for(i = 0; i < sps_num_ver_virtual_boundaries; i++)		
sps_virtual_boundaries_pos_x[i]	u(13)	
sps_num_hor_virtual_boundaries	u(2)	
for(i = 0; i < sps_num_hor_virtual_boundaries; i++)		
sps_virtual_boundaries_pos_y[i]	u(13)	
}		
if(sps_ptl_dpb_hrd_params_present_flag) {		
sps_general_hrd_params_present_flag	u(1)	
if(sps_general_hrd_params_present_flag) {		
general_hrd_parameters()		
if(sps_max_sublayers_minus1 > 0)		
sps_sublayer_cpb_params_present_flag	u(1)	
firstSubLayer = sps_sublayer_cpb_params_present_flag ? 0 :		
sps_max_sublayers_minus1		
ols_hrd_parameters(firstSubLayer, sps_max_sublayers_minus1)		
}		
}		
field_seq_flag	u(1)	
vui_parameters_present_flag	u(1)	
if(vui_parameters_present_flag)		
vui_parameters() /* Specified in ITU-T H.SEI ISO/IEC 23002-7 */		
sps_extension_flag	u(1)	
if(sps_extension_flag)		
while(more_rbsp_data())		
sps_extension_data_flag	u(1)	
rbsp_trailing_bits()		
}		

10

20

30

【 0 3 2 5】

40

50

【表 2 1 A】

1.1.1.2 ピクチャパラメータセットRBSP 構文

pic_parameter_set_rbsp() {	記述子
pps_pic_parameter_set_id	ue(v)
pps_seq_parameter_set_id	u(4)
pic_width_in_luma_samples	ue(v)
pic_height_in_luma_samples	ue(v)
conformance_window_flag	u(1)
if(conformance_window_flag) {	
conf_win_left_offset	ue(v)
conf_win_right_offset	ue(v)
conf_win_top_offset	ue(v)
conf_win_bottom_offset	ue(v)
}	
scaling_window_flag	u(1)
if(scaling_window_flag) {	
scaling_win_left_offset	ue(v)
scaling_win_right_offset	ue(v)
scaling_win_top_offset	ue(v)
scaling_win_bottom_offset	ue(v)
}	
output_flag_present_flag	u(1)
mixed_nalu_types_in_pic_flag	u(1)
pps_subpic_id_signalling_present_flag	u(1)
if(pps_subpics_id_signalling_present_flag) {	
pps_num_subpics_minus1	ue(v)
pps_subpic_id_len_minus1	ue(v)
for(i = 0; i <= pps_num_subpic_minus1; i++)	
pps_subpic_id[i]	u(v)
}	
no_pic_partition_flag	u(1)
if(!no_pic_partition_flag) {	
pps_log2_ctu_size_minus5	u(2)
num_exp_tile_columns_minus1	ue(v)
num_exp_tile_rows_minus1	ue(v)
for(i = 0; i <= num_exp_tile_columns_minus1; i++)	
tile_column_width_minus1[i]	ue(v)
for(i = 0; i <= num_exp_tile_rows_minus1; i++)	
tile_row_height_minus1[i]	ue(v)
rect_slice_flag	u(1)
if(rect_slice_flag)	
single_slice_per_subpic_flag	u(1)

10

20

30

40

50

【表 2 1 B】

if(rect_slice_flag && !single_slice_per_subpic_flag) {	
num_slices_in_pic_minus1	ue(v)
tile_idx_delta_present_flag	u(1)
for(i = 0; i < num_slices_in_pic_minus1; i++) {	
slice_width_in_tiles_minus1[i]	ue(v)
slice_height_in_tiles_minus1[i]	ue(v)
if(slice_width_in_tiles_minus1[i] == 0 && slice_height_in_tiles_minus1[i] == 0) {	
num_slices_in_tile_minus1[i]	ue(v)
numSlicesInTileMinus1 = num_slices_in_tile_minus1[i]	
for(j = 0; j < numSlicesInTileMinus1; j++)	
slice_height_in_ctu_minus1[i++]	ue(v)
}	
if(tile_idx_delta_present_flag && i < num_slices_in_pic_minus1)	
tile_idx_delta[i]	se(v)
}	
}	
loop_filter_across_tiles_enabled_flag	u(1)
loop_filter_across_slices_enabled_flag	u(1)
}	
entropy_coding_sync_enabled_flag	u(1)
if(!no_pic_partition_flag entropy_coding_sync_enabled_flag)	
entry_point_offsets_present_flag	u(1)
cabac_init_present_flag	u(1)
for(i = 0; i < 2; i++)	
num_ref_idx_default_active_minus1[i]	ue(v)
rpl1_idx_present_flag	u(1)
init_qp_minus26	se(v)
log2_transform_skip_max_size_minus2	ue(v)
cu_qp_delta_enabled_flag	u(1)
pps_cb_qp_offset	se(v)
pps_cr_qp_offset	se(v)
pps_joint_cbr_qp_offset_present_flag	u(1)
if(pps_joint_cbr_qp_offset_present_flag)	
pps_joint_cbr_qp_offset_value	se(v)
pps_slice_chroma_qp_offsets_present_flag	u(1)
pps_cu_chroma_qp_offset_list_enabled_flag	u(1)
if(pps_cu_chroma_qp_offset_list_enabled_flag) {	
chroma_qp_offset_list_len_minus1	ue(v)
for(i = 0; i <= chroma_qp_offset_list_len_minus1; i++) {	
cb_qp_offset_list[i]	se(v)
cr_qp_offset_list[i]	se(v)

10

20

30

40

50

【表 2 1 C】

if(pps_joint_cbr_qp_offset_present_flag)	
joint_cbr_qp_offset_list[i]	se(v)
}	
}	
pps_weighted_pred_flag	u(1)
pps_weighted_bipred_flag	u(1)
alf_present_in_ph_flag	u(1)
deblocking_filter_control_present_flag	u(1)
if(deblocking_filter_control_present_flag) {	
deblocking_filter_override_enabled_flag	u(1)
pps_deblocking_filter_disabled_flag	u(1)
if(!pps_deblocking_filter_disabled_flag) {	
pps_beta_offset_div2	se(v)
pps_tc_offset_div2	se(v)
}	
}	
constant_slice_header_params_enabled_flag	u(1)
if(constant_slice_header_params_enabled_flag) {	
pps_dep_quant_enabled_idc	u(2)
for(i = 0; i < 2; i++)	
pps_ref_pic_list_sps_idc[i]	u(2)
pps_mvd_l1_zero_idc	u(2)
pps_collocated_from_l0_idc	u(2)
pps_six_minus_max_num_merge_cand_plus1	ue(v)
pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1	ue(v)
}	
picture_header_extension_present_flag	u(1)
slice_header_extension_present_flag	u(1)
pps_extension_flag	u(1)
if(pps_extension_flag)	
while(more_rbsp_data())	
pps_extension_data_flag	u(1)
rbsp_trailing_bits()	
}	

【 0 3 2 6 】

10

20

30

40

50

【表 2 2 A】

一般的な制約情報構文

general_constraint_info() {	記述子
general_progressive_source_flag	u(1)
general_interlaced_source_flag	u(1)
general_non_packed_constraint_flag	u(1)
general_frame_only_constraint_flag	u(1)
intra_only_constraint_flag	u(1)
max_bitdepth_constraint_idc	u(4)
max_chroma_format_constraint_idc	u(2)
frame_only_constraint_flag	u(1)
no_qtbt_dual_tree_intra_constraint_flag	u(1)
no_partition_constraints_override_constraint_flag	u(1)
no_sao_constraint_flag	u(1)
no_alf_constraint_flag	u(1)
no_ccalf_constraint_flag	u(1)
no_joint_cbr_constraint_flag	u(1)
no_ref_wraparound_constraint_flag	u(1)
no_temporal_mvp_constraint_flag	u(1)
no_sbtmvp_constraint_flag	u(1)
no_amvr_constraint_flag	u(1)
no_bdof_constraint_flag	u(1)
no_dmvr_constraint_flag	u(1)
no_cclm_constraint_flag	u(1)
no_mts_constraint_flag	u(1)
no_sbt_constraint_flag	u(1)
no_affine_motion_constraint_flag	u(1)
no_bcw_constraint_flag	u(1)
no_ibc_constraint_flag	u(1)
no_ciip_constraint_flag	u(1)
no_fpel_mmvd_constraint_flag	u(1)
no_triangle_constraint_flag	u(1)
no_ladf_constraint_flag	u(1)
no_transform_skip_constraint_flag	u(1)
no_bdpcm_constraint_flag	u(1)
no_qp_delta_constraint_flag	u(1)
no_dep_quant_constraint_flag	u(1)
no_sign_data_hiding_constraint_flag	u(1)
no_mixed_nalu_types_in_pic_constraint_flag	u(1)
no_trail_constraint_flag	u(1)
no_stsa_constraint_flag	u(1)

10

20

30

40

50

【表 2 2 B】

no_rasl_constraint_flag	u(1)
no_radl_constraint_flag	u(1)
no_idr_constraint_flag	u(1)
no_cra_constraint_flag	u(1)
no_gdr_constraint_flag	u(1)
no_aps_constraint_flag	u(1)
while(!byte_aligned())	
gci_alignment_zero_bit	f(1)
num_reserved_constraint_bytes	u(8)
for(i = 0; i < num_reserved_constraint_bytes; i++)	
gci_reserved_constraint_byte[i]	u(8)
}	

10

【 0 3 2 7 】

【表 2 3】

7.3.3.2 一般的な制約情報構文

general_constraint_info() {	記述子
....	
no_alf_constraint_flag	u(1)
if (!no_alf_constraint_flag)	
no_ccalf_constraint_flag	u(1)

20

【 0 3 2 8 】

注:

!(0)=1

!(1)=0

【 0 3 2 9 】

30

40

50

【表 2 4 A】

7.3.2.4 ピクチャパラメータセット RBSP 構文
ピクチャヘッダ RBSP 構文

picture_header_rbsp() {	記述子
non_reference_picture_flag	u(1)
gdr_pic_flag	u(1)
no_output_of_prior_pics_flag	u(1)
if(gdr_pic_flag)	
recovery_poc_cnt	ue(v)
ph_pic_parameter_set_id	ue(v)
if(sps_poc_msb_flag) {	
ph_poc_msb_present_flag	u(1)
if(ph_poc_msb_present_flag)	
poc_msb_val	u(v)
}	
if(sps_subpic_id_present_flag && !sps_subpic_id_signalling_flag)	
{	
ph_subpic_id_signalling_present_flag	u(1)
if(ph_subpics_id_signalling_present_flag) {	
ph_subpic_id_len_minus1	ue(v)
for(i = 0; i <= sps_num_subpics_minus1; i++)	
ph_subpic_id[i]	u(v)
}	
}	
if(!sps_loop_filter_across_virtual_boundaries_disabled_present_flag)	
{	
ph_loop_filter_across_virtual_boundaries_disabled_present_flag	u(1)
if(ph_loop_filter_across_virtual_boundaries_disabled_present_flag)	
{	
ph_num_ver_virtual_boundaries	u(2)
for(i = 0; i < ph_num_ver_virtual_boundaries; i++)	
ph_virtual_boundaries_pos_x[i]	u(13)
ph_num_hor_virtual_boundaries	u(2)
for(i = 0; i < ph_num_hor_virtual_boundaries; i++)	
ph_virtual_boundaries_pos_y[i]	u(13)
}	
}	
}	
if(separate_colour_plane_flag == 1)	
colour_plane_id	u(2)
if(output_flag_present_flag)	
pic_output_flag	u(1)
pic_rpl_present_flag	u(1)

10

20

30

40

50

【表 2 4 B】

if(pic_rpl_present_flag) {	
for(i = 0; i < 2; i++) {	
if(num_ref_pic_lists_in_sps[i] > 0 && !pps_ref_pic_list_sps_idc[i] && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_sps_flag[i]	u(1)
if(pic_rpl_sps_flag[i]) {	
if(num_ref_pic_lists_in_sps[i] > 1 && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_idx[i]	u(v)
} else	
ref_pic_list_struct(i, num_ref_pic_lists_in_sps[i])	
for(j = 0; j < NumLtrpEntries[i][RplIdx[i]]; j++) {	
if(ltrp_in_slice_header_flag[i][RplIdx[i]])	
pic_poc_lsb_ltrp[i][j]	u(v)
pic_delta_poc_msb_present_flag[i][j]	u(1)
if(pic_delta_poc_msb_present_flag[i][j])	
pic_delta_poc_msb_cycle_ltrp[i][j]	ue(v)
}	
}	
}	
if(partition_constraints_override_enabled_flag) {	
partition_constraints_override_flag	u(1)
if(partition_constraints_override_flag) {	
pic_log2_diff_min_qt_min_cb_intra_slice_luma	ue(v)
pic_log2_diff_min_qt_min_cb_inter_slice	ue(v)
pic_max_mtt_hierarchy_depth_inter_slice	ue(v)
pic_max_mtt_hierarchy_depth_intra_slice_luma	ue(v)
if(pic_max_mtt_hierarchy_depth_intra_slice_luma != 0) {	
pic_log2_diff_max_bt_min_qt_intra_slice_luma	ue(v)
pic_log2_diff_max_tt_min_qt_intra_slice_luma	ue(v)
}	
if(pic_max_mtt_hierarchy_depth_inter_slice != 0) {	
pic_log2_diff_max_bt_min_qt_inter_slice	ue(v)
pic_log2_diff_max_tt_min_qt_inter_slice	ue(v)
}	
if(qtbtt_dual_tree_intra_flag) {	
pic_log2_diff_min_qt_min_cb_intra_slice_chroma	ue(v)
pic_max_mtt_hierarchy_depth_intra_slice_chroma	ue(v)
if(pic_max_mtt_hierarchy_depth_intra_slice_chroma != 0) {	
pic_log2_diff_max_bt_min_qt_intra_slice_chroma	ue(v)
pic_log2_diff_max_tt_min_qt_intra_slice_chroma	ue(v)

10

20

30

40

50

【表 2 4 C】

}	
}	
}	
}	
if(cu_qp_delta_enabled_flag) {	
pic_cu_qp_delta_subdiv_intra_slice	ue(v)
pic_cu_qp_delta_subdiv_inter_slice	ue(v)
}	
if(pps_cu_chroma_qp_offset_list_enabled_flag) {	
pic_cu_chroma_qp_offset_subdiv_intra_slice	ue(v)
pic_cu_chroma_qp_offset_subdiv_inter_slice	ue(v)
}	
if(sps_temporal_mvp_enabled_flag)	
pic_temporal_mvp_enabled_flag	u(1)
if(!pps_mvd_l1_zero_idc)	
mvd_l1_zero_flag	u(1)
if(!pps_six_minus_max_num_merge_cand_plus1)	
pic_six_minus_max_num_merge_cand	ue(v)
if(sps_affine_enabled_flag)	
pic_five_minus_max_num_subblock_merge_cand	ue(v)
if(sps_fpel_mmvd_enabled_flag)	
pic_fpel_mmvd_enabled_flag	u(1)
if(sps_bdof_pic_present_flag)	
pic_disable_bdof_flag	u(1)
if(sps_dmvr_pic_present_flag)	
pic_disable_dmvr_flag	u(1)
if(sps_prof_pic_present_flag)	
pic_disable_prof_flag	u(1)
if(sps_triangle_enabled_flag && MaxNumMergeCand >= 2 && !pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1)	
pic_max_num_merge_cand_minus_max_num_triangle_cand	ue(v)
if (sps_ibc_enabled_flag)	
pic_six_minus_max_num_ibc_merge_cand	ue(v)
if(sps_joint_cbr_enabled_flag)	
pic_joint_cbr_sign_flag	u(1)
if(sps_sao_enabled_flag) {	
pic_sao_enabled_present_flag	u(1)
if(pic_sao_enabled_present_flag) {	
pic_sao_luma_enabled_flag	u(1)
if(ChromaArrayType != 0)	
pic_sao_chroma_enabled_flag	u(1)

10

20

30

40

50

【表 2 4 D】

}	
}	
if(sps_alf_enabled_flag && alf_present_in_ph_flag) {	
pic_alf_enabled_present_flag	u(1)
if(pic_alf_enabled_present_flag) {	
pic_alf_enabled_flag	u(1)
if(pic_alf_enabled_flag) {	
pic_num_alf_aps_ids_luma	u(3)
for(i = 0; i < pic_num_alf_aps_ids_luma; i++)	
pic_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
pic_alf_chroma_idc	u(2)
if(pic_alf_chroma_idc)	
pic_alf_aps_id_chroma	u(3)
}	
if(ChromaArrayType != 0)	
if(sps_ccalf_enabled_flag) {	
pic_cross_component_alf_cb_enabled_flag	u(1)
if(pic_cross_component_alf_cb_enabled_flag) {	
pic_cross_component_alf_cb_aps_id	u(3)
pic_cross_component_cb_filters_signalled_minus1	ue(v)
}	
if(ChromaArrayType != 0)	u(1)
pic_cross_component_alf_cr_enabled_flag	
if(pic_cross_component_alf_cr_enabled_flag) {	
pic_cross_component_alf_cr_aps_id	u(3)
pic_cross_component_cr_filters_signalled_minus1	ue(v)
}	
}	
}	
}	
if (!pps_dep_quant_enabled_flag)	
pic_dep_quant_enabled_flag	u(1)
if(!pic_dep_quant_enabled_flag)	
sign_data_hiding_enabled_flag	u(1)
if(deblocking_filter_override_enabled_flag) {	
pic_deblocking_filter_override_present_flag	u(1)
if(pic_deblocking_filter_override_present_flag) {	
pic_deblocking_filter_override_flag	u(1)
if(pic_deblocking_filter_override_flag) {	
pic_deblocking_filter_disabled_flag	u(1)

10

20

30

40

50

【表 2 4 E】

if(!pic_deblocking_filter_disabled_flag) {	
pic_beta_offset_div2	se(v)
pic_tc_offset_div2	se(v)
}	
}	
}	
}	
if(sps_lmcs_enabled_flag) {	
pic_lmcs_enabled_flag	u(1)
if(pic_lmcs_enabled_flag) {	
pic_lmcs_aps_id	u(2)
if(ChromaArrayType != 0)	
pic_chroma_residual_scale_flag	u(1)
}	
}	
if(sps_scaling_list_enabled_flag) {	
pic_scaling_list_present_flag	u(1)
if(pic_scaling_list_present_flag)	
pic_scaling_list_aps_id	u(3)
}	
if(picture_header_extension_present_flag) {	
ph_extension_length	ue(v)
for(i = 0; i < ph_extension_length; i++)	
ph_extension_data_byte[i]	u(8)
}	
rbsp_trailing_bits()	
}	

10

20

30

【 0 3 3 0】

【表 2 5】

pic_parameter_set_rbsp() {	記述子
....	
if (sps_alf_enabled_flag)	
alf_present_in_ph_flag	u(1)
deblocking_filter_control_present_flag	u(1)
if(deblocking_filter_control_present_flag) {	
.....	

40

【 0 3 3 1】

1に等しいalf_present_in_ph_flagは、ALFの使用を有効にするための構文要素がPPSを参照するPH(ピクチャヘッダ)内に存在し得ることを指定する。0に等しいalf_present_in_ph_flagは、ALFの使用を有効にするための構文要素がPPSを参照するスライスヘッダ内に存在し得ることを指定する。

【 0 3 3 2】

50

【表 2 6】

7.3.2.6 ピクチャヘッダ RBSP 構文

picture_header_rbsp() {	記述子
.....	
if(sps_alf_enabled_flag && alf_present_in_ph_flag) {	
pic_alf_enabled_flag	u(1)
if(pic_alf_enabled_flag) {	
pic_num_alf_aps_ids_luma	u(3)
for(i = 0; i < pic_num_alf_aps_ids_luma; i++)	
pic_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
pic_alf_chroma_idc	u(2)
if(pic_alf_chroma_idc)	
pic_alf_aps_id_chroma	u(3)
if(sps_ccalf_enabled_flag) {	
pic_cross_component_alf_cb_enabled_flag	u(1)
if(pic_cross_component_alf_cb_enabled_flag) {	
pic_cross_component_alf_cb_aps_id	u(3)
}	
pic_cross_component_alf_cr_enabled_flag	u(1)
if(pic_cross_component_alf_cr_enabled_flag) {	
pic_cross_component_alf_cr_aps_id	u(3)
}	
} /* end of sps_ccalf_enabled_flag */	
} /* end of pic_alf_enabled_flag */	
} /* end of (sps_alf_enabled_flag && alf_present_in_ph_flag) */	
.....	

10

20

30

【0 3 3 3】

上記の表において規定されているように、(pic_cross_component_alf_cb_enabled_flagのような)第4の構文要素および(たとえば、pic_cross_component_alf_cr_enabled_flagのような)第7の構文要素のようなピクチャヘッダ内のCCALFに属するさらなる構文要素の存在は、第2の構文要素の存在および/または値に依存し得る。次いで、pic_cross_component_alf_cb_aps_idのような第5の構文要素の存在は、第2の構文要素、ならびに/または第4の構文要素の存在および/もしくは値に依存し得、pic_cross_component_alf_cr_aps_idのような第8の構文要素の存在および/または値は、第2の構文要素の存在および/もしくは値、ならびに/または第7の構文要素の存在および/もしくは値に依存し得る。

40

【0 3 3 4】

50

【表 2 7】

7.3.7.1 一般的なスライスヘッダ構文

slice_header() {	記述子
.....	
if(sps_alf_enabled_flag && !alf_present_in_ph_flag) {	
slice_alf_enabled_flag	u(1)
if(slice_alf_enabled_flag) {	
slice_num_alf_aps_ids_luma	u(3)
for(i = 0; i < slice_num_alf_aps_ids_luma; i++)	
slice_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
slice_alf_chroma_idc	u(2)
if(slice_alf_chroma_idc)	
slice_alf_aps_id_chroma	u(3)
if(sps_ccalf_enabled_flag) {	
slice_cross_component_alf_cb_enabled_flag	u(1)
if(slice_cross_component_alf_cb_enabled_flag) {	
slice_cross_component_alf_cb_aps_id	u(3)
}	
slice_cross_component_alf_cr_enabled_flag	u(1)
if(slice_cross_component_alf_cr_enabled_flag) {	
slice_cross_component_alf_cr_aps_id	u(3)
}	
} /* end of sps_ccalf_enabled_flag */	
} /* end of slice_alf_enabled_flag */	
} /* end of (sps_alf_enabled_flag && !alf_present_in_ph_flag) */	
.....	

10

20

30

【0 3 3 5】

上記の表において、CCALFに属するさらなる構文要素は、第2の構文要素の存在および/または値にさらに依存し得る。たとえば、slice_cross_component_alf_cb_enabled_flagのような第10の構文要素が、第2の構文要素に応じて提供され得る。この第10の構文要素ならびに/または第2の構文要素の存在および値に応じて、slice_cross_component_alf_cb_aps_idのような第11の構文要素が提供され得る。さらに、slice_cross_component_alf_cr_enabled_flagのような第12の構文要素が、第2の構文要素の値に応じてスライスヘッダにおいて提供され得る。第12の構文要素の存在および/もしくは値に応じて、ならびに/または第2の構文要素に応じて、slice_cross_component_alf_cr_aps_idのような第13の構文要素がスライスヘッダにおいて提供され得る。

40

【0 3 3 6】

注:

!(0) = 1

!(1) = 0

if(sps_alf_enabled_flag && !alf_present_in_ph_flag)は“if sps_alf_enabled_flag is true (and) if alf_present_in_ph_flag is false)”を意味する。

【0 3 3 7】

50

【表 2 8 A】

スライスヘッダ構文
一般的なスライスヘッダ構文

slice_header() {	記述子
slice_pic_order_cnt_lsb	u(v)
if(subpics_present_flag)	
slice_subpic_id	u(v)
if(rect_slice_flag NumTilesInPic > 1)	
slice_address	u(v)
if(!rect_slice_flag && NumTilesInPic > 1)	
num_tiles_in_slice_minus1	ue(v)
slice_type	ue(v)
if(!pic_rpl_present_flag && (nal_unit_type != IDR_W_RADL && nal_unit_type != IDR_N_LP) sps_idr_rpl_present_flag) {	
for(i = 0; i < 2; i++) {	
if(num_ref_pic_lists_in_sps[i] > 0 && !pps_ref_pic_list_sps_idc[i] && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
slice_rpl_sps_flag[i]	u(1)
if(slice_rpl_sps_flag[i]) {	
if(num_ref_pic_lists_in_sps[i] > 1 && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
slice_rpl_idx[i]	u(v)
} else	
ref_pic_list_struct(i, num_ref_pic_lists_in_sps[i])	
for(j = 0; j < NumLtrpEntries[i][RplIdx[i]]; j++) {	
if(ltrp_in_slice_header_flag[i][RplIdx[i]])	
slice_poc_lsb_ltrp[i][j]	u(v)
slice_delta_poc_msb_present_flag[i][j]	u(1)
if(slice_delta_poc_msb_present_flag[i][j])	
slice_delta_poc_msb_cycle_ltrp[i][j]	ue(v)
}	
}	
}	
}	
if(pic_rpl_present_flag ((nal_unit_type != IDR_W_RADL && nal_unit_type != IDR_N_LP) sps_idr_rpl_present_flag)) {	
if((slice_type != I && num_ref_entries[0][RplIdx[0]] > 1) (slice_type == B && num_ref_entries[1][RplIdx[1]] > 1)) {	

10

20

30

40

50

【表 2 8 B】

num_ref_idx_active_override_flag	u(1)
if(num_ref_idx_active_override_flag)	
for(i = 0; i < (slice_type == B ? 2 : 1); i++)	
if(num_ref_entries[i][RplIdx[i]] > 1)	
num_ref_idx_active_minus1[i]	ue(v)
}	
}	
if(slice_type != I) {	
if(cabac_init_present_flag)	
cabac_init_flag	u(1)
if(pic_temporal_mvp_enabled_flag) {	
if(slice_type == B && !pps_collocated_from_I0_idc)	
collocated_from_I0_flag	u(1)
if((collocated_from_I0_flag && NumRefIdxActive[0] > 1) (!collocated_from_I0_flag && NumRefIdxActive[1] > 1))	
collocated_ref_idx	ue(v)
}	
if((pps_weighted_pred_flag && slice_type == P) (pps_weighted_bipred_flag && slice_type == B))	
pred_weight_table()	
}	
slice_qp_delta	se(v)
if(pps_slice_chroma_qp_offsets_present_flag) {	
slice_cb_qp_offset	se(v)
slice_cr_qp_offset	se(v)
if(sps_joint_cbr_enabled_flag)	
slice_joint_cbr_qp_offset	se(v)
}	
if(pps_cu_chroma_qp_offset_list_enabled_flag)	
cu_chroma_qp_offset_enabled_flag	u(1)
if(sps_sao_enabled_flag && !pic_sao_enabled_present_flag) {	
slice_sao_luma_flag	u(1)
if(ChromaArrayType != 0)	
slice_sao_chroma_flag	u(1)
}	
if(sps_alf_enabled_flag && !alf_present_in_ph_flag pic_alf_enabled_present_flag) {	
slice_alf_enabled_flag	u(1)
if(slice_alf_enabled_flag) {	
slice_num_alf_aps_ids_luma	u(3)
for(i = 0; i < slice_num_alf_aps_ids_luma; i++)	
slice_alf_aps_id_luma[i]	u(3)

10

20

30

40

50

【表 2 8 C】

if(ChromaArrayType != 0)	
slice_alf_chroma_idc	u(2)
if(slice_alf_chroma_idc)	
slice_alf_aps_id_chroma	u(3)
}	
if(sps_ccalf_enabled_flag) { if(ChromaArrayType != 0)	
slice_cross_component_alf_cb_enabled_flag	u(1)
if(slice_cross_component_alf_cb_enabled_flag) {	
slice_cross_component_alf_cb_aps_id	u(3)
slice_cross_component_cb_filters_signalled_minus1	ue(v)
}	
if(ChromaArrayType != 0)	
slice_cross_component_alf_cr_enabled_flag	u(1)
if(slice_cross_component_alf_cr_enabled_flag) {	
slice_cross_component_alf_cr_aps_id	u(3)
slice_cross_component_cr_filters_signalled_minus1	ue(v)
}	
}	
}	
}	
if(deblocking_filter_override_enabled_flag && !pic_deblocking_filter_override_present_flag)	
slice_deblocking_filter_override_flag	u(1)
if(slice_deblocking_filter_override_flag) {	
slice_deblocking_filter_disabled_flag	u(1)
if(!slice_deblocking_filter_disabled_flag) {	
slice_beta_offset_div2	se(v)
slice_tc_offset_div2	se(v)
}	
}	
if(entry_point_offsets_present_flag && NumEntryPoints > 0) {	
offset_len_minus1	ue(v)
for(i = 0; i < NumEntryPoints; i++)	
entry_point_offset_minus1[i]	u(v)
}	
if(slice_header_extension_present_flag) {	
slice_header_extension_length	ue(v)
for(i = 0; i < slice_header_extension_length; i++)	
slice_header_extension_data_byte[i]	u(8)
}	
byte_alignment()	
}	

10

20

30

40

【0 3 3 8】

上記の表において、いくつかの要素が取り消し線として示されている。これは、第1の実施形態において、これらの要素が存在し得ることを意味する。代替実施形態において、これらの要素は、キャンセルされ得、残りの構文をそのまま残し得る。たとえば、要素ChromaArrayTypeは、提供されない場合がある。これに対応して、他の構文要素の条件付き存在も含む、この要素からの任意の依存関係が、第1の実施形態において提供され得る。代替実施形態において、この依存関係は、存在せず、結果として、たとえば、いずれの場合においても構文要素が存在するが、代替実施形態において、それらは、ChromaArrayTypeが特定の値を有する場合にのみ存在した。

50

【0339】

同様に、slice_cross_component_cr_filters_signalled_minus1およびslice_cross_component_cb_filters_signalled_minus1のような構文要素は、(ChromaArrayTypeのような構文要素が存在するかどうかに関係なく)まったく提供されない場合がある。

【0340】

新しく導入された構文要素のセマンティクスは、以下のとおりである。

【0341】

0に等しい(sps_ccalf_enabled_flag)のような第2の構文要素の値は、クロスコンポーネント適応ループフィルタが無効にされていることを指定する。1に等しい第1の構文要素の値は、クロスコンポーネント適応ループフィルタが有効にされていることを指定する。これは、逆にも提供され得、すなわち、第2の構文要素が1に等しい値を有する場合、CCALFが無効にされ得、第2の構文要素が0に等しい値を有する場合、CCALFが有効にされ得る。

【0342】

1に等しいno_ccalf_constraint_flagは、第1の構文要素(sps_ccalf_enabled_flagなど)が0に等しいべきであることを指定する。0に等しいno_ccalf_constraint_flagは、そのような制約を課さない。

【0343】

実施形態において、フィルタリングプロセスについて、以下に詳細に提示する。

【0344】

上記で論じた実施形態について、特定の構文要素の意味と、特定の値を取るこれらの構文要素の結果とに関して、いくつかの一般的な説明を以下で行う。以下に提示する開示は、上記の実施形態のいずれか、具体的には代替実施形態1から5によって包含されることを意図している。

【0345】

8.8 ループ内フィルタプロセス

8.8.1 一般

【0346】

4. (sps_alf_enabled_flagとして上記で示した)第1の構文要素が1に等しい場合、以下が適用される。

- 第2の構文要素(sps_ccalf_enabled_flagなど)が1に等しい場合、再構築されたピクチャサンプル配列(適応ループフィルタの前) S_L' は、再構築されたピクチャサンプル配列 S_L に等しく設定される。

- 節8.8.5.1において指定されているような適応ループフィルタプロセスは、再構築されたピクチャサンプル配列 S_L で呼び出され、第14の構文要素(上記の表において、これは、ChromaArrayTypeとして示されている)が0に等しくない場合、入力として配列 S_{cb} および S_{cr} 、および修正された再構築されたピクチャサンプル配列 S_L' であり、そして第14の構文要素が0に等しくない場合、出力として、適応ループフィルタ後の配列 S'_{cb} および S'_{cr} である。

- 配列 S_L' 、そして第14の構文要素が0に等しくない場合、配列 S'_{cb} および S'_{cr} が配列 S_L に割り当てられ、第14の構文要素が0に等しくない場合、それぞれ、配列 S_{cb} および S_{cr} (復号されたピクチャを表す)である。

- 第2の構文要素(sps_ccalf_enabled_flagなど)が1に等しい場合、以下が適用される。

- 節x.x.x.xにおいて指定されているクロスコンポーネント適応ループフィルタは、再構築されたピクチャサンプル配列 S_L で呼び出され、第14の構文要素(ChromaArrayTypeなど)が0に等しくない場合、入力として配列 S_{cb} および S_{cr} 、ならびに修正された再構築されたピクチャサンプル配列 S_L' 、第14の構文要素が0に等しくない場合、出力としてクロスコンポーネント適応ループフィルタ後の配列 S'_{cb} および S'_{cr} である。

- 配列 S_L' 、そして第14の構文要素が0に等しくない場合、配列 S'_{cb} および S'_{cr} が配列 S_L に割り当てられ、第14の構文要素が0に等しくない場合、それぞれ、配列 S_{cb} および S_{cr}

10

20

30

40

50

c_r (復号されたピクチャを表す)である。

【0347】

本発明は、ここで提示した代替案に限定されず、むしろ、非常に一般的な方法における本発明は、スライスヘッダにおいて存在するCCALFデータがピクチャヘッダにおいて条件付きでシグナリングされることを可能にすることが留意されるべきである。ピクチャヘッダにおいてCCALFを使用する主な利点は、スライスヘッダにおける信号オーバーヘッドが低減することである。

【0348】

本発明は、スライスヘッダにおいてシグナリングされる特定のCCALF構文要素が(たとえば、いくつかのビットストリーム適合要件のために)すべてのスライスにわたって同じ値を有する場合、その特定の構文要素がもはやスライスヘッダにおいてシグナリングされず、むしろすべてのスライスに関連付けられたピクチャヘッダにおいて1回だけシグナリングされる場合もカバーする。

【0349】

クロスコンポーネント適応ループフィルタ(CC-ALF)は、それぞれ、ループフィルタおよびポスト処理ステップとして使用され得る。

【0350】

CC-ALFは、各クロマ成分を精緻化するためにルマサンプル値を使用する。CC-ALFは、クロマサンプルフィルタリングのための補正係数を導出するためにルマサンプルに適用される。

【0351】

JVET-P0080およびJVET-00630は、クロスコンポーネントALFフィルタと呼ばれる新しいループ内フィルタを提案している。CC-ALFは、適応ループフィルタプロセスの一部として動作し、各クロマ成分(C_r または C_b 成分)を精緻化するためにルマサンプル値を利用する。ツールは、ビットストリーム内の情報によって制御され、この情報は、(a)各クロマ成分に関するフィルタ係数と、(b)サンプルのブロックに対するフィルタの適用を制御するマスクの両方を含む。フィルタ係数は、適応パラメータセット(APS(Adaptation parameter set))においてシグナリングされ、ブロックサイズおよびマスクは、スライスレベルにおいてシグナリングされる。

【0352】

マスクは、ビット(0または1)であることが理解され得る。ここで、マスクは、サンプルのブロックがフィルタリングされるべきかどうかを示し、(マスク=1は、フィルタリングされることを意味し)および(マスク=0は、フィルタリングされないことを意味する)。基本的に、APSは、ALFに関する係数と他の情報とを担持する。

【0353】

CC-ALFの一般的な設計を、図6(a)、図6(b)、および図6(c)において示す。この一般的な設計は、CC-ALF関連の問題に言及する上記の実施形態のいずれかに適用され得る。具体的には、ここで説明することは、代替実施形態1から5の各々によって包含されることを意図している。

【0354】

フィルタの配置は、図6(a)において示すようになる(左側を参照)。フィルタの形状は、図6(b)および図6(c)において示すようになる。

【0355】

CC-ALFは、線形のダイヤモンド形フィルタ(図6(b)および図6(c))を各クロマ成分に関するルマチャネルに適用することによって動作し、これは、

【0356】

【数1】

10

20

30

40

50

$$\Delta I_i(x, y) = \sum_{(x_0, y_0) \in S_i} I_0(x_c + x_0, y_c + y_0) c_i(x_0, y_0)$$

【0357】

として表され、ここで、

(x, y)は、精緻化されているクロマ成分iの場所であり、

(x_c, y_c)は、(x, y)に基づくルマの場所であり、

S_iは、クロマ成分iに関するルマにおけるフィルタサポートであり、

c_i(x₀, y₀)は、フィルタ係数を表す。

【0358】

クロマ成分iについてi=1の場合、第1のクロマ成分(cb色成分など)が存在し、クロマ成分iについてi=2の場合、第2のクロマ成分(cr色成分など)が存在することが留意される。

【0359】

CC-ALFの主な特徴は、以下を含む。

- ・ サポート領域を中心とするルマの場所(x_c, y_c)は、ルマ平面とクロマ平面との間の空間スケーリング係数に基づいて計算される。基本的に、CCALFは、クロマサンプルを精緻化するためにルマサンプルを使用する。したがって、フィルタは、ルマサンプルに適用され、次いで、クロマのための補正係数が導出される。

- ・ すべてのフィルタ係数は、APSにおいて送信され、8ビットのダイナミックレンジを有する。

- ・ APSは、スライスヘッダにおいて参照され得る。

- ・ スライスの各クロマ成分に対して使用されるCC-ALF係数はまた、時間的サブレイヤに対応するバッファ内に記憶される。時間的サブレイヤフィルタ係数のこれらのセットの再利用は、スライスレベルのフラグを使用して容易になる。時間的サブレイヤは、基本的に参照情報に基づいて定義されることが理解され得る。

- ・ CC-ALFフィルタの適用は、可変ブロックサイズにおいて制御され、サンプルの各ブロックについて受信されるコンテキストコーディングされたフラグによってシグナリングされる。ブロックサイズとCC-ALF有効化フラグとは、各クロマ成分についてスライスレベルにおいて受信される。

- ・ 水平仮想境界に対する境界パディングは、繰り返しを利用する。残りの境界について、通常のALFと同じタイプのパディングが使用される。

【0360】

図8は、図7と比較して、共通のCCALFデータを定義する、CCALFのためのピクチャヘッダエントリが導入され、次いで、すべてのスライスがこの共通情報を継承することができることを示すブロック図である。

【0361】

図16は、データ構造5000の概略図であり、データ構造5000は、図2におけるエンコーダによって生成され、図3におけるデコーダによって受信されるビットストリーム21の一部を表し得る。データ構造5000(すなわち、ビデオビットストリーム5000)は、VPS5005と、SPS5010と、PPS5015と、少なくとも4つのスライス5020、5025、5030、および5035とを備え得る。スライス5020は、ヘッダ5040とデータ5045とを備え得る。スライス5025、5030、5035は、スライス5020と同様である。データ5045は、少なくとも3つのブロック5050、5055、および5060、すなわち、少なくとも3つのブロック5050、5055、および5060のコーディングされた表現を含み得る。4つのスライス5020～5035が示されているが、ビデオビットストリーム5000は、任意の適切な数のスライスを含む。3つのブロック5050～5060が示されているが、データ5045は、任意の適切な数

10

20

30

40

50

のブロックを含む。それに加えて、残りの各スライス5025、5030、5035も、ブロックを含む。したがって、ビデオビットストリーム5000は、すなわち、多数のブロックのコーディングされた表現を含むが、ビデオビットストリーム5000は、1つのSPS5010と、ブロックよりも著しく少ないスライスヘッダとを含む。複数のCC-ALF関連構文要素を含むビットストリームは、図16においても示され得ることが理解され得る。上記および下記の構文によって示されているように、複数のCC-ALF関連構文要素が明確に指定されている。それに加えて、ビットストリームを符号化または復号するためのプロセスは、上記および下記に示す構文および/またはセマンティクスにおいて明確に指定されている。

【0362】

ここに示すビットストリームは、上記で説明した実施形態のいずれかを使用してビデオシーケンスを符号化することによって取得され得る。具体的には、このビットストリームは、取得され得、または上記の代替実施形態1から5において説明したように、構文要素、具体的にはCC-ALFに関連する構文要素を含み得る。

【0363】

図14は、たとえば、ビデオを符号化する方法4200のフローチャートを示す。この方法は、たとえば、図12によるエンコーダを使用して実施され得るが、本明細書で開示されている任意の他のエンコーダおよび符号化デバイスも、この方法を実行するために使用され得る。

【0364】

符号化の方法4200は、クロマ成分を精緻化するためにクロスコンポーネント適応ループフィルタCC-ALFを適用するステップ4201と、複数のCC-ALF関連構文要素を含むビットストリームを生成するステップ4202であって、複数のCC-ALF関連構文要素がCC-ALF関連情報を示す、ステップ4202とを含む。ステップ4202において生成されたビットストリーム内で、複数のCC-ALF関連構文要素は、シーケンスパラメータセット(SPS)レベル、ピクチャヘッダ、またはスライスヘッダのうちの任意の1つまたは複数においてシグナリングされる。複数のCC-ALF関連構文要素は、第1の構文要素であって、クロスコンポーネント適応ループフィルタを含む適応ループフィルタ(ALF)がシーケンスレベルにおいて有効にされているかどうかを示し、SPSレベルにおいてシグナリングされる、第1の構文要素と、第2の構文要素であって、クロスコンポーネント適応ループフィルタがシーケンスレベルにおいて有効にされているかどうかを示し、SPSレベルにおいてシグナリングされる、第2の構文要素とを含む。

【0365】

図15は、たとえば、ビットストリームなどからのビデオの復号の方法4300のフローチャートを示す。この方法4300は、たとえば、図13によるデコーダを使用して実施され得る。方法4300を実行するために、本明細書で開示されている任意の他のデコーダおよび復号デバイスも使用され得る。

【0366】

方法4300は、ビットストリームからの複数のクロスコンポーネント適応ループフィルタCC-ALF関連構文要素を解析するステップ4301であって、複数のCC-ALF関連構文要素が、シーケンスパラメータセット(SPS)レベル、ピクチャヘッダ、またはスライスヘッダのうちの任意の1つまたは複数から取得される、ステップ4301を含む。複数のCC-ALF関連構文要素は、第1の構文要素であって、クロスコンポーネント適応ループフィルタを含む適応ループフィルタ(ALF)がシーケンスレベルにおいて有効にされているかどうかを示し、SPSレベルにおいてシグナリングされる、第1の構文要素と、第2の構文要素であって、クロスコンポーネント適応ループフィルタがシーケンスレベルにおいて有効にされているかどうかを示し、SPSレベルにおいてシグナリングされる、第2の構文要素とを含む。方法4300は、複数のCC-ALF関連構文要素のうちの少なくとも1つを使用してCC-ALFプロセスを実行するステップ4302をさらに含む。

【0367】

一実施形態において、符号化デバイスによって実施される符号化の方法が提供され、方

10

20

30

40

50

法は、

クロスコンポーネント適応ループフィルタ(CC-ALF)を適用することによってフィルタリングプロセス(クロスコンポーネントフィルタリングプロセスなど)を実行するステップと、

複数のCC-ALF関連構文要素(M個のCC-ALF関連構文要素などであり、 $M \geq 1$ であり、Mは、整数である)を含むビットストリームを生成するステップであって、複数のCC-ALF関連構文要素がCC-ALF関連情報を示す、ステップと

を含み、

複数のCC-ALF関連構文要素が、ビデオパラメータセット(VPS)レベル、シーケンスパラメータセット(SPS)レベル、ピクチャパラメータセット(PPS)レベル、ピクチャヘッダ、スライスヘッダ、もしくはタイルヘッダのうちの任意の1つもしくは複数においてシグナリングされ、または複数のCC-ALF関連構文要素が、シーケンスパラメータセット(PPS)レベルおよび/もしくはピクチャヘッダにおいてシグナリングされる。

【0368】

この方法では、たとえば、ピクチャのすべてのスライスに関連する情報が一回だけピクチャヘッダにおいて提供され得るので、サイズが低減されたビットストリームが提供され得る。

【0369】

一実施形態において、複数のCC-ALF関連構文要素は、第1の構文要素(たとえば、`sps_ccalf_enabled_flag`または`sps_alf_enabled_flag`などの第1のインジケータ)を含み、第1の構文要素(たとえば、`sps_ccalf_enabled_flag`などの第1のインジケータ)は、シーケンスパラメータセット(PPS)レベルにおいてシグナリングされる。このインジケータは、ALFまたはCC-ALFまたはその両方がシーケンス全体に対して有効にされ得るかどうを示し得る。

【0370】

さらなる実施形態において、第1の構文要素(たとえば、`sps_ccalf_enabled_flag`などの第1のインジケータ)は、クロスコンポーネント適応ループフィルタ(CC-ALF)がシーケンスレベルにおいて有効にされているかどうかを示し、または

第1の構文要素(たとえば、`sps_alf_enabled_flag`などの第1のインジケータ)は、クロスコンポーネント適応ループフィルタ(CC-ALF)を含む適応ループフィルタ(ALF)がシーケンスレベルにおいて有効にされているかどうかを示す。この第1の構文要素により、適応ループフィルタのモードは、具体的には、すべてのピクチャまたはシーケンスの一部についてシーケンスレベルにおいて提供され得る。

【0371】

第1の構文要素(`sps_ccalf_enabled_flag`または`sps_alf_enabled_flag`など)が第1の値(真または1)として設定されるか、または第1の構文要素が第1の値を有する(真または1)ことがさらに規定され得る。この値により、好ましくは単一のビット値のみを使用して、ALFおよび/またはCC-ALFが有効にされているかどうかを示され得る。

【0372】

一実施形態において、複数のCC-ALF関連構文要素は、

第1の構文要素(`sps_ccalf_enabled_flag`または`sps_alf_enabled_flag`など)が第1の値(真または1)として設定されているか、もしくは第1の構文要素が第1の値(真または1)を有する場合、第2の構文要素(たとえば、`pic_ccalf_enabled_present_flag`などの第2のインジケータ)であって、第2構文要素(`pic_ccalf_enabled_present_flag`など)がピクチャヘッダにおいてシグナリングされる第2の構文要素か、または

第1の構文要素(`sps_ccalf_enabled_flag`または`sps_alf_enabled_flag`など)が第1の値(真または1)として設定されているか、もしくは第1の構文要素が第1の値(真または1)を有する場合、第9の構文要素(たとえば、`pic_alf_enabled_present_flag`などの第9のインジケータ)であって、第9の構文要素(`pic_alf_enabled_present_flag`など)がピクチャヘッダにおいてシグナリングされる、第9の構文要素をさらに含む。

【0373】

第1の構文要素が第1の値として設定されている場合に第2の構文要素が提供されること(およびそれに対応して第9の構文要素について)は、それぞれの構文要素が、第1の構文要素が第1の値を有する場合にのみ、または少なくともその場合に提供され、他の場合には提供されないことを意味すると理解され得る。それにもかかわらず、この実施形態は、第2および第9の構文要素が第1の構文要素の値に関係なく提供されるまたは存在することを包含し得る。この意味は、別の構文要素が特定の値を取る場合にこれらの構文要素が存在すると言及されるという条件で、本明細書で言及されているすべてのさらなる構文要素の存在または非存在に適用され得る。

【0374】

第2および第9の構文要素により、低減したビット量を使用して、ALFおよび/またはCC-ALFがピクチャのスライスに対して有効にされているかどうか指定され得る。

【0375】

一実施形態において、第2の構文要素(たとえば、pic_ccalf_enabled_present_flagなどの第2のインジケータ)は、第3の構文要素および/または構文要素の第4のセットがピクチャヘッダ内に存在するかどうかを示す。

【0376】

複数のCC-ALF関連構文要素は、

第2の構文要素(pic_ccalf_enabled_present_flagなど)または第9の構文要素(pic_alf_enabled_present_flagなど)が第1の値(真または1)として設定されているか、または第2の構文要素または第9の構文要素が第1の値(真または1)を有する場合、第3の構文要素(たとえば、pic_ccalf_enabled_flagまたはpic_alf_enabled_flagなどの第3のインジケータ)であって、第3の構文要素(pic_ccalf_enabled_flagまたはpic_alf_enabled_flagなど)がピクチャヘッダにおいてシグナリングされる、第3の構文要素をさらに含むことがさらに規定され得る。

【0377】

さらなる実施形態において、第3の構文要素(たとえば、pic_ccalf_enabled_flagなどの第3のインジケータ)は、クロスコンポーネント適応ループフィルタ(CCALF)がピクチャヘッダに関連付けられたすべてのスライスに対して有効にされているかどうかを示すか、または

第3の構文要素(たとえば、pic_alf_enabled_flagなどの第3のインジケータ)は、クロスコンポーネント適応ループフィルタ(CCALF)を含む適応ループフィルタがピクチャヘッダに関連付けられたすべてのスライスに対して有効にされているかどうかを示す。これによって、ALFまたはCC-ALFの適用に関する情報は、低減した情報量を使用して、ピクチャのすべてのスライスに関してシグナリングされ得る。

【0378】

さらなる実施形態において、複数のCC-ALF関連構文要素は、

第3の構文要素(たとえば、pic_ccalf_enabled_flagまたはpic_alf_enabled_flagなどの第3のインジケータ)が第1の値(真または1)として設定されているか、または第3の構文要素が第1の値(真または1)を有する場合、構文要素の第4のセットであって、構文要素の第4のセットがピクチャヘッダにおいて(たとえば、ピクチャヘッダレベルにおいて)シグナリングされるか、または構文要素の第4のセットがピクチャヘッダエントリによって担持される、構文要素の第4のセットをさらに含む。

【0379】

具体的には、一実施形態において、構文要素の第4のセットは、以下を含む。

【0380】

10

20

30

40

50

【表 2 9】

if(ChromaArrayType != 0)	
pic_cross_component_alf_cb_enabled_flag	u(1)
if(pic_cross_component_alf_cb_enabled_flag) {	
pic_cross_component_alf_cb_aps_id	u(3)
}	
if(ChromaArrayType != 0)	
pic_cross_component_alf_cr_enabled_flag	u(1)
if(pic_cross_component_alf_cr_enabled_flag) {	
pic_cross_component_alf_cr_aps_id	u(3)
}	
}	

10

【 0 3 8 1】

構文要素の第4のセットが以下を含むことが、追加的または代替的に規定され得る。

【 0 3 8 2】

20

【表 3 0】

if(ChromaArrayType != 0)	
pic_cross_component_alf_cb_enabled_flag	u(1)
if(pic_cross_component_alf_cb_enabled_flag) {	
pic_cross_component_alf_cb_aps_id	u(3)
pic_cross_component_cb_filters_signalled_minus1	ue(v)
}	
if(ChromaArrayType != 0)	
pic_cross_component_alf_cr_enabled_flag	u(1)
if(pic_cross_component_alf_cr_enabled_flag) {	
pic_cross_component_alf_cr_aps_id	u(3)
pic_cross_component_cr_filters_signalled_minus1	ue(v)
}	
}	

30

【 0 3 8 3】

40

さらに、追加的または代替的に、構文要素の第4のセットは、以下を含む。

【 0 3 8 4】

50

【表 3 1】

if(ChromaArrayType != 0)	
pic_cross_component_alf_cb_enabled_flag	u(1)
if(pic_cross_component_alf_cb_enabled_flag) {	
pic_cross_component_cb_filters_signalled_minus1	ue(v)
}	
if(ChromaArrayType != 0)	
pic_cross_component_alf_cr_enabled_flag	u(1)
if(pic_cross_component_alf_cr_enabled_flag) {	
pic_cross_component_cr_filters_signalled_minus1	ue(v)
}	

10

【0385】

上記の実施形態では、適用されるべきフィルタに関する関連情報が、すでにピクチャヘッダ内にあるすべてのスライスについて提供され得、ビットストリームのサイズを潜在的に低減する。

20

【0386】

さらなる実施形態において、同じピクチャのすべてのスライスについて、同じCC-ALF関連情報(aps_idsなど)は、ピクチャヘッダから継承されることが規定される。したがって、ピクチャヘッダにおいて提供される情報は、スライスヘッダなどの中に追加で含まれる必要がなく、それによって、ビットストリームにおいて提供される情報量を潜在的に低減する。

【0387】

一実施形態において、複数のCC-ALF関連構文要素は、

第1の構文要素(たとえば、sps_ccalf_enabled_flagまたはsps_alf_enabled_flagなどの第1のインジケータ)であって、第1の構文要素(たとえば、sps_ccalf_enabled_flagなどの第1のインジケータ)がシーケンスパラメータセット(SPS)レベルにおいてシグナリングされる、第1の構文要素と、

30

第1の構文要素(sps_ccalf_enabled_flagまたはsps_alf_enabled_flagなど)が第1の値(真または1)として設定されているか、または第1の値(真または1)を有する場合、第2の構文要素(たとえば、pic_ccalf_enabled_present_flagなどの第2のインジケータ)であって、第2の構文要素(pic_ccalf_enabled_present_flagなど)がピクチャヘッダにおいてシグナリングされる、第2の構文要素とを含む。

【0388】

さらに、複数のCC-ALF関連構文要素は、

40

第1の構文要素(sps_ccalf_enabled_flagまたはsps_alf_enabled_flagなど)が第1の値(真または1)として設定されているか、または第1の値(真または1)を有する場合、および第2の構文要素(pic_ccalf_enabled_present_flagまたはpic_alf_enabled_present_flagなど)が第2の値(偽または0)として設定されているか、または第2の値(偽または0)を有する場合、第5の構文要素(たとえば、slice_ccalf_enabled_flagまたはslice_alf_enabled_flagなどの第5のインジケータ)であって、第5の構文要素(slice_ccalf_enabled_flagまたはslice_alf_enabled_flagなど)がスライスヘッダにおいてシグナリングされる、第5の構文要素を含み得る。

【0389】

複数のCC-ALF関連構文要素は、

50

第5の構文要素(たとえば、`slice_ccalf_enabled_flag`または`slice_alf_enabled_flag`などの第5のインジケータ)が第1の値(真または1)として設定されているか、または第5の構文要素が第1の値(真または1)を有する場合、構文要素の第6のセットであって、構文要素の第6のセットがスライスヘッダにおいてシグナリングされるか、または構文要素の第6のセットがスライスヘッダエントリによって担持される、構文要素の第6のセットをさらに含むことも規定され得る。構文要素のこのセットは、たとえば、ALFまたはCC-ALFにおいて使用されるべきパラメータを示す。第5の構文要素の値に潜在的に依存するこのセットを提供することによって、ビットストリームのサイズは、さらに低減され得る。

【0390】

具体的には、構文要素の第6のセットは、以下を含むことが規定され得る。

【0391】

【表32】

if(ChromaArrayType != 0)	
slice_cross_component_alf_cb_enabled_flag	u(1)
if(slice_cross_component_alf_cb_enabled_flag) {	
slice_cross_component_alf_cb_aps_id	u(3)
}	
if(ChromaArrayType != 0)	
slice_cross_component_alf_cr_enabled_flag	u(1)
if(slice_cross_component_alf_cr_enabled_flag) {	
slice_cross_component_alf_cr_aps_id	u(3)
}	
}	

【0392】

追加的または代替的に、構文要素の第6のセットは、以下を含むことが規定され得る。

【0393】

【表33】

if(ChromaArrayType != 0)	
slice_cross_component_alf_cb_enabled_flag	u(1)
if(slice_cross_component_alf_cb_enabled_flag) {	
slice_cross_component_alf_cb_aps_id	u(3)
slice_cross_component_cb_filters_signalled_minus1	ue(v)
}	
if(ChromaArrayType != 0)	
slice_cross_component_alf_cr_enabled_flag	u(1)
if(slice_cross_component_alf_cr_enabled_flag) {	
slice_cross_component_alf_cr_aps_id	u(3)
slice_cross_component_cr_filters_signalled_minus1	ue(v)
}	

【0394】

さらに、構文要素の第6のセットは、以下を含むことが規定され得る。

【0395】

【表 3 4】

if(ChromaArrayType != 0)	
slice_cross_component_alf_cb_enabled_flag	u(1)
if(slice_cross_component_alf_cb_enabled_flag) {	
slice_cross_component_cb_filters_signalled_minus1	ue(v)
}	
if(ChromaArrayType != 0)	
slice_cross_component_alf_cr_enabled_flag	u(1)
if(slice_cross_component_alf_cr_enabled_flag) {	
slice_cross_component_cr_filters_signalled_minus1	ue(v)
}	

10

【0 3 9 6】

さらなる実施形態において、複数のCC-ALF関連構文要素は、

第1の構文要素(sps_ccalf_enabled_flagまたはsps_alf_enabled_flagなど)が第1の値(真または1)として設定されているか、または第1の値(真または1)を有する場合、第7の構文要素(たとえば、pic_cross_component_alf_cb_aps_idなどの第7のインジケータ)と第8の構文要素(たとえば、pic_cross_component_alf_cr_aps_idなどの第8のインジケータ)とをさらに含む。

20

【0 3 9 7】

一実施形態において、複数のCC-ALF関連構文要素は、

第1の構文要素(sps_ccalf_enabled_flagまたはsps_alf_enabled_flagなど)が第1の値(真または1)として設定されているか、または第1の値(真または1)を有する場合、および第9の構文要素(pic_alf_enabled_present_flagなど)が第1の値(真または1)として設定されているか、または第1の値(真または1)を有する場合、第7の構文要素(たとえば、pic_cross_component_alf_cb_aps_idなどの第7のインジケータ)と第8の構文要素(たとえば、pic_cross_component_alf_cr_aps_idなどの第8のインジケータ)とをさらに含む。

30

【0 3 9 8】

一実施形態において、複数のCC-ALF関連構文要素は、

第1の構文要素(sps_ccalf_enabled_flagまたはsps_alf_enabled_flagなど)が第1の値(真または1)に設定されているか、または第1の値(真または1)を有する場合、および第9の構文要素(pic_alf_enabled_present_flagなど)が第1の値(真または1)に設定されているか、または第1の値(真または1)を有する場合、第7の構文要素(たとえば、pic_cross_component_alf_cb_aps_idなどの第7のインジケータ)と、第8の構文要素(たとえば、pic_cross_component_alf_cr_aps_idなどの第8のインジケータ)と、第2の構文要素(pic_ccalf_enabled_present_flag)とをさらに含む。

40

【0 3 9 9】

複数のCC-ALF関連構文要素は、

第1の構文要素(sps_ccalf_enabled_flagまたはsps_alf_enabled_flagなど)が第1の値(真または1)に設定されているか、または第1の値(真または1)を有する場合、第7の構文要素(たとえば、pic_cross_component_alf_cb_aps_idなどの第7のインジケータ)と、第8の構文要素(たとえば、pic_cross_component_alf_cr_aps_idなどの第8のインジケータ)と、第9の構文要素(pic_alf_enabled_present_flag)とをさらに含むことも規定され得る。

【0 4 0 0】

50

さらに、一実施形態において、CC-ALFは、現在の画像ブロックのクロマ成分のクロマサンプル(各クロマサンプルなど)を精緻化するために、現在の画像ブロックのルマ成分の1つまたは複数のルマサンプルに対するフィルタリング処理を実行するために適用される。

【0401】

具体的には、CC-ALFは、ルマサンプル値を使用して各クロマ成分を精緻化するように構成されることが規定され得る。

【0402】

一実施形態において、CC-ALFは、適応ループフィルタプロセスの一部として動作する。

【0403】

画像ブロックは、ルマブロックとクロマブロックとを備え、

第1のクロマブロックは、画像ブロックの第1のクロマ成分(Cb成分など)であり、第2のクロマブロックは、画像ブロックの第2のクロマ成分(Cr成分など)であることも規定され得る。

【0404】

本開示は、

クロスコンポーネントALF(CC-ALF)が有効にされているかどうかを判断するステップと

、
1つまたは複数の構文要素(たとえば、pic_ccalf_enabled_flag、pic_cross_component_alf_cb_enabled_flag、またはpic_cross_component_alf_cr_enabled_flagなどのCC-ALF関連構文要素)を含むビットストリームを生成するステップであって、1つまたは複数の構文要素が、クロスコンポーネントALF(CC-ALF)がピクチャヘッダレベルおよび対応するCC-ALF関連情報において有効にされているかどうかを示し、構文要素がピクチャヘッダレベルにおいてシグナリングされる、ステップと

を含む、

符号化デバイスによって実施される符号化の方法にさらに関する。それによって、後の復号のための関連情報は、ビットストリームにおいて低減したサイズで提供され得る。

【0405】

CC-ALFは、現在の画像ブロックのクロマ成分のクロマサンプル(各クロマサンプルなど)を精緻化するために、現在の画像ブロックのルマ成分の1つまたは複数のルマサンプルに対するフィルタリング処理を実行するために適用されることも規定され得る。

【0406】

一実施形態において、方法は、

CC-ALFフィルタを使用することによって、ピクチャに属する現在の画像ブロックのルマ成分に対してフィルタリング処理を実行するステップであって、現在の画像ブロックのルマ成分の1つまたは複数のルマサンプルが、現在の画像ブロックのクロマ成分の少なくとも1つのクロマサンプル(各クロマサンプルなど)を精緻化するために使用される、ステップをさらに含む。

【0407】

CC-ALFは、ルマサンプル値を使用して各クロマ成分を精緻化するように構成されることも規定され得る。

【0408】

さらなる実施形態において、CC-ALFは、適応ループフィルタプロセスの一部として動作する。

【0409】

画像ブロックは、ルマブロックとクロマブロックとを備え、

第1のクロマブロックは、画像ブロックの第1のクロマ成分(Cb成分など)であり、第2のクロマブロックは、画像ブロックの第2のクロマ成分(Cr成分など)であることも規定され得る。

【0410】

本開示は、復号デバイスによって実施される復号の方法にもさらに関し、方法は、

ビデオ信号のビットストリームからの1つまたは複数の構文要素を解析するステップであって、1つまたは複数の構文要素がクロスコンポーネント適応ループフィルタ(CC-ALF)関連情報を示し、1つもしくは複数の構文要素がビットストリームのビデオパラメータセット(VPS)レベル、シーケンスパラメータセット(SPS)レベル、ピクチャパラメータセット(PPS)レベル、ピクチャヘッダ、スライスヘッダ、もしくはタイルヘッダのうちの任意の1つもしくは複数から取得されるか、または1つもしくは複数の構文要素がシーケンスパラメータセット(SPS)レベルおよび/もしくはピクチャヘッダから取得される、ステップと、構文要素に基づいて、もしくは構文要素の値に基づいてCC-ALFを適用することによってフィルタリングプロセス(クロスコンポーネントフィルタリングプロセスなど)を実行するステップを含む。

10

【0411】

適用されるべきフィルタリング方法と、それがどのように適用されるべきかとを決定するために必要であり得る情報は、信頼性の高い復号を依然として可能にしながら、低減したサイズを有するビットストリームにおいてこれを提供され得る。

【0412】

本開示は、復号デバイスによって実施される復号の方法についてさらに説明し、方法は、ビデオ信号のビットストリームからの複数の構文要素を解析するステップであって、構文要素が、ビットストリームのビデオパラメータセット(VPS)レベル、シーケンスパラメータセット(SPS)レベル、ピクチャパラメータセット(PPS)レベル、ピクチャヘッダ、スライスヘッダ、またはタイルヘッダのうちの任意の1つもしくは複数から取得されるか、または構文要素がシーケンスパラメータセット(SPS)レベルおよび/もしくはピクチャヘッダから取得される、ステップと、

20

複数の構文要素からの1つまたは複数の構文要素に基づいてクロスコンポーネント適応ループフィルタ(CC-ALF)関連情報を決定するステップと、

CC-ALF関連情報に基づいてCC-ALFを適用することによってフィルタリングプロセス(クロスコンポーネントフィルタリングプロセスなど)を実行するステップとを含む。

【0413】

適用されるべきフィルタリング方法と、それがどのように適用されるべきかとを決定するために必要であり得る情報は、信頼性の高い復号を依然として可能にしながら、低減したサイズを有するビットストリームにおいてこれを提供され得る。

30

【0414】

1つもしくは複数の構文要素または複数の構文要素(CC-ALF関連構文要素など)は、第1の構文要素(たとえば、sps_ccalf_enabled_flagまたはsps_alf_enabled_flagなどの第1のインジケータ)を含み、第1の構文要素(sps_ccalf_enabled_flagなどの第1のインジケータ)は、シーケンスパラメータセット(SPS)レベルから取得されることが規定され得る。

【0415】

具体的には、第1の構文要素(たとえば、sps_ccalf_enabled_flagなどの第1のインジケータ)が、クロスコンポーネント適応ループフィルタ(CC-ALF)がシーケンスレベルにおいて有効にされているかどうかを示すか、または

40

第1の構文要素(たとえば、sps_alf_enabled_flagなどの第1のインジケータ)が、クロスコンポーネント適応ループフィルタ(CC-ALF)を含む適応ループフィルタ(ALF)がシーケンスレベルにおいて有効にされているかどうかを示すことがさらに規定され得る。

【0416】

それによって、どのフィルタ方法が適用されるべきかに関する情報が、ビットストリームにおいて低減した情報量でデコーダに提供され得る。

【0417】

50

一実施形態において、第1の構文要素(sps_ccalf_enabled_flagまたはsps_alf_enabled_flagなど)は、第1の値(真または1)として導出されるか、または第1の構文要素は、第1の値(真または1)を有する。

【0418】

複数の構文要素(CC-ALF関連構文要素など)は、

第1の構文要素(sps_ccalf_enabled_flagまたはsps_alf_enabled_flagなど)が第1の値(真または1)として導出されるか、もしくは第1の構文要素が第1の値(真または1)を有する場合、第2の構文要素(たとえば、pic_ccalf_enabled_present_flagなどの第2のインジケータ)であって、第2の構文要素(pic_ccalf_enabled_present_flagなど)がピクチャヘッダから取得される、第2の構文要素、または

10

第1の構文要素(sps_ccalf_enabled_flagまたはsps_alf_enabled_flagなど)が第1の値(真または1)として導出されるか、もしくは第1の構文要素が第1の値(真または1)を有する場合、第9の構文要素(たとえば、pic_alf_enabled_present_flagなどの第9のインジケータ)であって、第9の構文要素(pic_alf_enabled_present_flagなど)がピクチャヘッダから取得される、第9の構文要素をさらに含むことも規定され得る。

【0419】

さらなる実施形態において、第2の構文要素(たとえば、pic_ccalf_enabled_present_flagなどの第2のインジケータ)は、第3の構文要素、および/または構文要素の第4のセットがピクチャヘッダ内に存在するかどうかを示す。

20

【0420】

複数の構文要素(CC-ALF関連構文要素など)は、

第2の構文要素(pic_ccalf_enabled_present_flagなど)もしくは第9の構文要素(pic_alf_enabled_present_flagなど)が第1の値(真または1)として導出されるか、または第2の構文要素もしくは第9の構文要素が第1の値(真または1)を有する場合、第3の構文要素(たとえば、pic_ccalf_enabled_flagまたはpic_alf_enabled_flagなどの第3のインジケータ)をさらに含み得、第3の構文要素(pic_ccalf_enabled_flagまたはpic_alf_enabled_flagなど)は、ピクチャヘッダから取得される。

【0421】

具体的には、第3の構文要素(たとえば、pic_ccalf_enabled_flagなどの第3のインジケータ)が、クロスコンポーネント適応ループフィルタ(CCALF)がピクチャヘッダに関連付けられたすべてのスライスに対して有効にされているかどうかを示すか、または

30

第3の構文要素(たとえば、pic_alf_enabled_flagなどの第3のインジケータ)が、クロスコンポーネント適応ループフィルタ(CCALF)を含む適応ループフィルタがピクチャヘッダに関連付けられたすべてのスライスに対して有効にされているかどうかを示すことが規定され得る。

【0422】

一実施形態において、複数の構文要素(CC-ALF関連構文要素など)は、

第3の構文要素(たとえば、pic_ccalf_enabled_flagまたはpic_alf_enabled_flagなどの第3のインジケータ)が第1の値(真または1)として導出されるか、または第3の構文要素が第1の値(真または1)を有する場合、構文要素の第4のセットをさらに含み、構文要素の第4のセットは、ピクチャヘッダから(たとえば、ピクチャヘッダレベルにおいて)取得されるか、または構文要素の第4のセットは、ピクチャヘッダエントリによって担持される。

40

【0423】

構文要素のこのセットは、第3の構文要素が第1の値を取るか、または第1の値として導出される場合にのみ提供され得、そうでない場合には存在しない場合があるか、または構文要素のセットの値は、第3の構文要素が異なる値を取る場合、0のようなデフォルト値に設定され得る。両方の場合において、復号のための関連情報の信頼できる提供がビットストリームの低減されたサイズにおいて保証され得る。

【0424】

50

具体的には、構文要素の第4のセットは、以下を含み得る。

【0 4 2 5】

【表 3 5】

if(ChromaArrayType != 0)	
pic_cross_component_alf_cb_enabled_flag	u(1)
if(pic_cross_component_alf_cb_enabled_flag) {	
pic_cross_component_alf_cb_aps_id	u(3)
}	
if(ChromaArrayType != 0)	
pic_cross_component_alf_cr_enabled_flag	u(1)
if(pic_cross_component_alf_cr_enabled_flag) {	
pic_cross_component_alf_cr_aps_id	u(3)
}	
}	

10

【0 4 2 6】

代替的または追加的に、構文要素の第4のセットは、以下を含み得る。

20

【0 4 2 7】

【表 3 6】

if(ChromaArrayType != 0)	
pic_cross_component_alf_cb_enabled_flag	u(1)
if(pic_cross_component_alf_cb_enabled_flag) {	
pic_cross_component_alf_cb_aps_id	u(3)
pic_cross_component_cb_filters_signalled_minus1	ue(v)
}	
if(ChromaArrayType != 0)	
pic_cross_component_alf_cr_enabled_flag	u(1)
if(pic_cross_component_alf_cr_enabled_flag) {	
pic_cross_component_alf_cr_aps_id	u(3)
pic_cross_component_cr_filters_signalled_minus1	ue(v)
}	
}	

30

【0 4 2 8】

さらなる実施形態において、構文要素の第4のセットは、以下を含み得る。

40

【0 4 2 9】

50

【表 3 7】

if(ChromaArrayType != 0)	
pic_cross_component_alf_cb_enabled_flag	u(1)
if(pic_cross_component_alf_cb_enabled_flag) {	
pic_cross_component_cb_filters_signalled_minus1	ue(v)
}	
if(ChromaArrayType != 0)	
pic_cross_component_alf_cr_enabled_flag	u(1)
if(pic_cross_component_alf_cr_enabled_flag) {	
pic_cross_component_cr_filters_signalled_minus1	ue(v)
}	

10

【0 4 3 0】

同じピクチャのすべてのスライスについて、同じCC-ALF関連情報(aps_idsなど)がピクチャヘッダから継承されることが規定され得る。これによって、ピクチャのスライスに関するCC-ALF関連情報をデコーダに示すためにビットストリームにおいて提供される情報の量が低減され得る。

20

【0 4 3 1】

一実施形態において、複数の構文要素(CC-ALF関連構文要素など)は、

第1の構文要素(たとえば、sps_ccalf_enabled_flagまたはsps_alf_enabled_flagなどの第1のインジケータ)であって、第1の構文要素(たとえば、sps_ccalf_enabled_flagなどの第1のインジケータ)がシーケンスパラメータセット(SPS)レベルから取得される、第1の構文要素と、

第1の構文要素(sps_ccalf_enabled_flagまたはsps_alf_enabled_flagなど)が第1の値(真または1)として導出されるか、または第1の値(真または1)を有する場合、第2の構文要素(たとえば、pic_ccalf_enabled_present_flagなどの第2のインジケータ)であって、第2の構文要素(pic_ccalf_enabled_present_flagなど)がピクチャヘッダから取得される、第2の構文要素とを含む。

30

【0 4 3 2】

さらなる実施形態において、複数の構文要素(CC-ALF関連構文要素など)は、

第1の構文要素(sps_ccalf_enabled_flagまたはsps_alf_enabled_flagなど)が第1の値(真または1)として導出されるか、もしくは第1の値(真または1)を有する場合、および第2の構文要素(pic_ccalf_enabled_present_flagまたはpic_alf_enabled_present_flagなど)が第2の値(偽または0)として導出されるか、もしくは第2の値(偽または0)を有する場合、第5の構文要素(たとえば、slice_ccalf_enabled_flagまたはslice_alf_enabled_flagなどの第5のインジケータ)をさらに含み、第5の構文要素(slice_ccalf_enabled_flagまたはslice_alf_enabled_flagなど)は、スライスヘッダから取得される。

40

【0 4 3 3】

複数の構文要素(CC-ALF関連構文要素など)は、

第5の構文要素(たとえば、slice_ccalf_enabled_flagまたはslice_alf_enabled_flagなどの第5のインジケータ)が第1の値(真または1)として導出されるか、または第5の構文要素が第1の値(真または1)を有する場合、構文要素の第6のセットをさらに含み、構文要素の第6のセットは、スライスヘッダから取得されるか、または構文要素の第6のセットは、スライスヘッダエントリによって担持されることが規定され得る。

【0 4 3 4】

50

具体的には、構文要素の第6のセットは、以下を含むことが規定され得る。

【0 4 3 5】

【表 3 8】

if(ChromaArrayType != 0)	
slice_cross_component_alf_cb_enabled_flag	u(1)
if(slice_cross_component_alf_cb_enabled_flag) {	
slice_cross_component_alf_cb_aps_id	u(3)
}	
if(ChromaArrayType != 0)	
slice_cross_component_alf_cr_enabled_flag	u(1)
if(slice_cross_component_alf_cr_enabled_flag) {	
slice_cross_component_alf_cr_aps_id	u(3)
}	
}	

10

【0 4 3 6】

代替的または追加的に、構文要素の第6のセットは、以下を含むことが規定され得る。

【0 4 3 7】

【表 3 9】

if(ChromaArrayType != 0)	
slice_cross_component_alf_cb_enabled_flag	u(1)
if(slice_cross_component_alf_cb_enabled_flag) {	
slice_cross_component_alf_cb_aps_id	u(3)
slice_cross_component_cb_filters_signalled_minus1	ue(v)
}	
if(ChromaArrayType != 0)	
slice_cross_component_alf_cr_enabled_flag	u(1)
if(slice_cross_component_alf_cr_enabled_flag) {	
slice_cross_component_alf_cr_aps_id	u(3)
slice_cross_component_cr_filters_signalled_minus1	ue(v)
}	

30

【0 4 3 8】

同様に、構文要素の第6のセットは、以下を含むことが規定され得る。

【0 4 3 9】

40

50

【表 4 0】

if(ChromaArrayType != 0)	
slice_cross_component_alf_cb_enabled_flag	u(1)
if(slice_cross_component_alf_cb_enabled_flag) {	
slice_cross_component_cb_filters_signalled_minus1	ue(v)
}	
if(ChromaArrayType != 0)	
slice_cross_component_alf_cr_enabled_flag	u(1)
if(slice_cross_component_alf_cr_enabled_flag) {	
slice_cross_component_cr_filters_signalled_minus1	ue(v)
}	

10

【 0 4 4 0】

一実施形態において、複数の構文要素(CC-ALF関連構文要素など)は、

第1の構文要素(sps_ccalf_enabled_flagまたはsps_alf_enabled_flagなど)が第1の値(真または1)として導出されるか、または第1の値(真または1)を有する場合、第7の構文要素(たとえば、pic_cross_component_alf_cb_aps_idなどの第7のインジケータ)と第8の構文要素(たとえば、pic_cross_component_alf_cr_aps_idなどの第8のインジケータ)とをさらに含む。

20

【 0 4 4 1】

複数の構文要素(CC-ALF関連構文要素など)は、

第1の構文要素(sps_ccalf_enabled_flagまたはsps_alf_enabled_flagなど)が第1の値(真または1)として導出されるか、または第1の値(真または1)を有する場合、および第9の構文要素(pic_alf_enabled_present_flagなど)が第1の値(真または1)として導出されるか、または第1の値(真または1)を有する場合、第7の構文要素(たとえば、pic_cross_component_alf_cb_aps_idなどの第7のインジケータ)と第8の構文要素(たとえば、pic_cross_component_alf_cr_aps_idなどの第8のインジケータ)とをさらに含むことも規定され得る。

30

【 0 4 4 2】

別の実施形態において、複数の構文要素(CC-ALF関連構文要素など)は、

第1の構文要素(sps_ccalf_enabled_flagまたはsps_alf_enabled_flagなど)が第1の値(真または1)として導出されるか、または第1の値(真または1)を有する場合、および第9の構文要素(pic_alf_enabled_present_flagなど)が第1の値(真または1)として導出されるか、または第1の値(真または1)を有する場合、第7の構文要素(たとえば、pic_cross_component_alf_cb_aps_idなどの第7のインジケータ)と、第8の構文要素(たとえば、pic_cross_component_alf_cr_aps_idなどの第8のインジケータ)と、第2の構文要素(pic_ccalf_enabled_present_flagなど)とをさらに含む。

40

【 0 4 4 3】

複数の構文要素(CC-ALF関連構文要素など)は、

第1の構文要素(sps_ccalf_enabled_flagまたはsps_alf_enabled_flagなど)が第1の値(真または1)として導出されるか、または第1の値(真または1)を有する場合、第7の構文要素(たとえば、pic_cross_component_alf_cb_aps_idなどの第7のインジケータ)と、第8の構文要素(たとえば、pic_cross_component_alf_cr_aps_idなどの第8のインジケータ)と、第9の構文要素(pic_alf_enabled_present_flagなど)とをさらに含むことも規定され得る。

【 0 4 4 4】

50

一実施形態において、CC-ALFは、現在の画像ブロックのクロマ成分のクロマサンプル(各クロマサンプルなど)を精緻化するために、現在の画像ブロックのルマ成分の1つまたは複数のルマサンプルに対してフィルタリングプロセスを実行するために適用される。

【0445】

CC-ALFは、ルマサンプル値を使用して各クロマ成分を精緻化するように構成され得る。

【0446】

CC-ALFは、適応ループフィルタプロセスの一部として動作することも規定され得る。

【0447】

一実施形態において、画像ブロックは、ルマブロックとクロマブロックとを備え、

第1のクロマブロックは、画像ブロックの第1のクロマ成分(Cb成分など)であり、第2のクロマブロックは、画像ブロックの第2のクロマ成分(Cr成分など)である。

【0448】

本開示は、

ビデオ信号のビットストリームから1つまたは複数の構文要素(たとえば、M個のCC-ALF関連構文要素であり、Mが整数であり、 $M \geq 1$ であり、pic_ccalf_enabled_flag、pic_cross_component_alf_cb_enabled_flag、またはpic_cross_component_alf_cr_enabled_flagなど)を解析するステップであって、1つまたは複数の構文要素が、クロスコンポーネントALF(CCALF)がピクチャヘッダレベルにおいて有効にされているかどうかと、対応するCCALF関連情報とを示し、1つまたは複数の構文要素がピクチャヘッダレベルから取得される、ステップと、

CC-ALFが有効にされているという判断に応答して、CC-ALFを適用することによってフィルタリングプロセス(クロスコンポーネントフィルタリングプロセスなど)を実行するステップと

を含む、復号デバイスによって実施される復号の方法にさらに言及する。

【0449】

この方法は、ビットストリームのサイズを低減することを可能にしながら、復号中にCC-ALFを適切に適用し得る。

【0450】

一実施形態において、CC-ALFは、現在の画像ブロックのクロマ成分のクロマサンプル(各クロマサンプルなど)を精緻化するために、現在の画像ブロックのルマ成分の1つまたは複数のルマサンプルに対してフィルタリング処理を実行するために適用される。

【0451】

CC-ALFは、ルマサンプル値を使用して各クロマ成分を精緻化するように構成されることも規定され得る。

【0452】

さらなる実施形態において、CC-ALFは、適応ループフィルタプロセスの一部として動作する。

【0453】

実施形態は、画像ブロックが、ルマブロックとクロマブロックとを備え、

第1のクロマブロックが画像ブロックの第1のクロマ成分(Cb成分など)であり、第2のクロマブロックが画像ブロックの第2のクロマ成分(Cr成分など)である、ことをさらに包含し得る。

【0454】

本開示は、ビデオデータを符号化するためのデバイスをさらに提供し、デバイスは、ビデオデータメモリと、ビデオエンコーダと

を備え、ビデオエンコーダは、

クロスコンポーネント適応ループフィルタ(CC-ALF)を適用することによってフィルタリングプロセス(クロスコンポーネントフィルタリングプロセス)を実行し、

複数の構文要素(CC-ALF関連構文要素など)を含めることによってビデオ信号のためのビ

10

20

30

40

50

ットストリームを生成する

ように構成され、複数の構文要素がCC-ALF関連情報を示し、

複数のCC-ALF関連構文要素が、ビデオパラメータセット(VPS)レベル、シーケンスパラメータセット(SPS)レベル、ピクチャパラメータセット(PPS)レベル、ピクチャヘッダ、スライスヘッダ、もしくはタイルヘッダのうちの任意の1つもしくは複数から取得されるか、または

複数のCC-ALF関連構文要素が、シーケンスパラメータセット(SPS)レベルおよび/もしくはピクチャヘッダから取得される。

【0455】

このデバイスは、低減したサイズを有するビットストリームを結果として生じる符号化を実現することができる。

【0456】

さらに、ビデオデータを復号するためのデバイスが提供され、デバイスは、

ビデオデータメモリと、

ビデオデコーダであって、ビデオデコーダが、

ビデオ信号のビットストリームからの複数の構文要素を解析することであって、複数の構文要素が、ビットストリームのビデオパラメータセット(VPS)レベル、シーケンスパラメータセット(SPS)レベル、ピクチャパラメータセット(PPS)レベル、ピクチャヘッダ、スライスヘッダ、またはタイルヘッダのうちの任意の1つもしくは複数から取得されるか、または複数の構文要素が、ビットストリームのシーケンスパラメータセット(SPS)レベル

および/もしくはピクチャヘッダから取得される、ことと、
複数の構文要素からの1つまたは複数の構文要素に基づいて、クロスコンポーネント適応ループフィルタ(CC-ALF)関連情報を決定することと、

CC-ALF関連情報に基づいてCC-ALFを適用することによって、フィルタリングプロセス(クロスコンポーネントフィルタリングプロセスなど)を実行すること
とを行うように構成された、ビデオデコーダ
とを備える。

【0457】

このデコーダは、復号中に適用されるべきフィルタに関する情報を取得するために、低減したサイズを有するビットストリームを必要としながら、信頼性の高い復号を実行することができる。

【0458】

本開示は、エンコーダをさらに提供し、エンコーダは、上記の実施形態のうちの任意の1つによる方法を実行するための処理回路を備える。

【0459】

本開示は、デコーダにも言及し、デコーダは、上記の実施形態のうちの任意の1つによる方法を実行するための処理回路を備える。

【0460】

さらに、上記の実施形態のうちの任意の1つによる方法を実行するためのプログラムコードを備えるコンピュータプログラム製品が提供される。

【0461】

本開示は、デコーダにさらに関し、デコーダは、

1つまたは複数のプロセッサと、

プロセッサに結合され、プロセッサによって実行するためのプログラミングを記憶する非一時的コンピュータ可読記憶媒体と
を備え、プログラミングは、プロセッサによって実行されると、上記の実施形態のうちのいずれか1つによる方法を実行するようにデコーダを構成する。

【0462】

さらに、本開示においてエンコーダが提供され、エンコーダは、

1つまたは複数のプロセッサと、

10

20

30

40

50

プロセッサに結合され、プロセッサによって実行するためのプログラミングを記憶する非一時的コンピュータ可読記憶媒体と
を備え、プログラミングは、プロセッサによって実行されると、上記の実施形態のうちのいずれか1つによる方法を実行するようにエンコーダを構成する。

【0463】

さらに、画像復号デバイスによって復号された符号化ビットストリームを含む非一時的記録媒体が提供され、ビットストリームが、ビデオ信号または画像信号のフレームを複数のブロックに分割することによって生成され、複数の構文要素(CC-ALF関連構文要素など)を含み、複数の構文要素がCC-ALF関連情報を示し、

複数の構文要素(CC-ALF関連構文要素など)が、ビデオパラメータセット(VPS)レベル、シーケンスパラメータセット(SPS)レベル、ピクチャパラメータセット(PPS)レベル、ピクチャヘッダ、スライスヘッダ、またはタイルヘッダのうちの任意の1つまたは複数から取得されるか、または

複数の構文要素(CC-ALF関連構文要素など)が、シーケンスパラメータセット(SPS)レベルまたはピクチャヘッダのうちの少なくとも1つから取得される。

【0464】

本開示は、装置にも関し、装置は、

1つまたは複数のプロセッサによって実行されると、ビデオビットストリームに対応する画像データを生成するための動作を1つまたは複数のプロセッサに実行させる命令をその上に記憶している非一時的コンピュータ可読媒体を備え、画像データが、

N個のスライスを含むピクチャを含むビデオビットストリーム内の複数のピクチャと、ピクチャのN個のスライスに関連付けられたピクチャヘッダとを含み、

N個のスライスのうちの1つもしくは複数または各々が、エントロピー符号化されたサンプルの複数のブロックを含み、

CC-ALFフィルタリング情報またはCC-ALFフィルタパラメータが、ピクチャヘッダからのN個のスライスによって継承されるか、またはN個のスライスが、ピクチャヘッダからのALFフィルタリングデータを継承する。

【0465】

上記の実施形態のいずれかに即して、`slice_cross_component_alf_cr_aps_id`および`slice_cross_component_alf_cb_aps_id`の値は、`pic_cross_component_alf_cb_aps_id`および`pic_cross_component_alf_cr_aps_id`と同じであることがさらに規定され得る。

【0466】

さらなる実施形態において、`pic_cross_component_alf_cb_aps_id`および`pic_cross_component_alf_cr_aps_id`は、ピクチャヘッダから取得される。

【0467】

CCALF関連構文要素が、N個のスライスすべてに関連付けられたピクチャヘッダにおいて一回だけ取得されることも規定され得る。

【0468】

一実施形態において、Nは、正の整数であり、1よりも大きい。

【0469】

一実施形態において、符号化デバイスによって実施されるビデオビットストリームの符号化のための方法が提供され、符号化方法は、

複数の構文要素(たとえば、`sps_alf_enabled_flag`などのALF関連構文要素、および/または`sps_ccalf_enabled_flag`などのCCALF関連構文要素)を含めることによってビデオ信号のためのビットストリームを生成するステップを含み、

複数の構文要素は、クロスコンポーネント適応ループフィルタ(CC-ALF)有効化フラグ(`sps_ccalf_enabled_flag`など)を含み、

CC-ALF関連パラメータは、少なくともCC-ALF有効化フラグ(`sps_ccalf_enabled_flg`

10

20

30

40

50

gなど)の値に基づいて、条件付きでシグナリングされる。

【0470】

これは、復号のための関連情報を含むが、低減したサイズを有するビットストリームを提供し得る。

【0471】

さらに、復号デバイスによって実施されるビデオビットストリームの復号のための方法が、本開示に即して提供され、復号方法は、

ビデオビットストリームから複数の構文要素を取得するステップ(S110)を含み、

複数の構文要素は、クロスコンポーネント適応ループフィルタ(CC-ALF)有効化フラグ(sps_ccalf_enabled_flagなど)を含み、

CC-ALF関連パラメータは、少なくともCCALF有効化フラグ(sps_ccalf_enabled_flagなど)の値に基づいて、条件付きでシグナリングされる。

【0472】

この方法は、ビデオビットストリームがサイズにおいて低減された場合であっても、信頼性の高い復号を実行することを可能にする。

【0473】

一実施形態において、1つまたは複数の条件が満たされる場合、CCALF有効化フラグ(sps_ccalf_enabled_flagなど)は、ビットストリームのシーケンスパラメータセット(SPS)レベルにおいてシグナリングされ、1つまたは複数の条件は、ビットストリームのシーケンスパラメータセット(SPS)レベルにおいてシグナリングされる第1のフラグ(sps_alf_enabled_flagなど)の値が有効化値(第1の値など、たとえば、tureまたは1)である場合を含み、

CCALF有効化フラグは、クロスコンポーネント適応ループフィルタ(CCALF)がシーケンスレベルまたはSPSレベルにおいて有効にされているかどうかを示す。

【0474】

構文要素は、第3のフラグ(alf_present_in_ph_flagなど)を含み、第3のフラグは、ビットストリームのピクチャパラメータセット(PPS)レベルでシグナリングされ、第3のフラグは、ALFの使用を有効にするための1つまたは複数の構文要素が、PPSを参照するPH(ピクチャヘッダ)内に存在するかどうかを示すことも規定され得る。

【0475】

第3のフラグ(alf_present_in_ph_flag)が1に等しいことは、ALFの使用を有効にするための構文要素が、PPSを参照するPH(ピクチャヘッダ)内に存在し得ることを指定することも規定され得る。0に等しい第3のフラグ(alf_present_in_ph_flag)は、ALFの使用を有効にするための構文要素が、PPSを参照するスライスヘッダ内に存在し得ることを指定する。

【0476】

一実施形態において、第1のフラグ(sps_alf_enabled_flagなど)の値が有効値(第1の値など、たとえば、tureまたは1)であり、第3のフラグ(alf_present_in_ph_flag)の値が有効値(第1の値など、たとえば、tureまたは1)である場合、および

CCALF有効化フラグ(sps_ccalf_enabled_flagなど)の値が有効値(第1の値など、たとえば、tureまたは1)である場合、クロスコンポーネント適応ループフィルタ(CCALF)関連パラメータ(CCALFの使用を有効にするための構文要素など)は、ビットストリームのピクチャヘッダレベルでシグナリングされることが規定される。

【0477】

さらなる実施形態において、第1のフラグ(sps_alf_enabled_flagなど)の値が有効値(第1の値など、たとえば、tureまたは1)であり、第3のフラグ(alf_present_in_ph_flag)の値が無効値(第2の値など、たとえば、偽または0)である場合、および

CCALF有効化フラグ(sps_ccalf_enabled_flagなど)の値が有効値(第1の値など、たとえば、真または1)である場合、クロスコンポーネント適応ループフィルタ(CCALF)関連パラメータ(CCALFの使用を有効にするための構文要素など)は、ビットストリームのスライ

10

20

30

40

50

スヘッダレベルでシグナリングされることが規定される。

【0478】

さらなる実施形態において、0に等しいCC-ALF有効化フラグ(sps_ccalf_enabled_flagなど)は、クロスコンポーネント適応ループフィルタが無効にされていることを指定し、1に等しいCC-ALF有効化フラグ(sps_ccalf_enabled_flagなど)は、クロスコンポーネント適応ループフィルタが有効にされていることを指定する、ことが規定される。

【0479】

構文要素は、

1に等しい第2のフラグ(no_ccalf_constraint_flagなど)が、CCALF有効化フラグ(sps_ccalf_enabled_flagなど)が0に等しいことを指定し、または0に等しい第2のフラグ(no_ccalf_constraint_flagなど)がそのような制約を課さないことを含むことがさらに規定され得る。

【0480】

一実施形態において、方法は、

フィルタリングされた再構築されたピクチャ(修正された再構築されたルマサンプル配列 S'_L 、修正された再構築されたクロマ配列 S'_{cb} 、および/または修正された再構築されたクロマサンプル配列 S'_{cr} など)を取得するために、再構築されたピクチャ(再構築されたルマサンプル配列 S_L 、再構築されたクロマサンプル配列 S_{cb} 、および/または再構築されたクロマサンプル配列 S_{cr})に対してフィルタリングプロセス(クロスコンポーネント適応ループフィルタプロセス)を実行するステップをさらに含み得る。

【0481】

本開示は、複数の構文要素(たとえば、sps_alf_enabled_flagなどのALF関連構文要素、および/またはsps_ccalf_enabled_flagなどのCCALF関連構文要素)を含めることによって、ビデオ信号のための符号化ビットストリームをさらに提供し、複数の構文要素は、クロスコンポーネント適応ループフィルタ(CCALF)有効化フラグ(sps_ccalf_enabled_flagなど)を含み、

CCALF関連パラメータは、少なくともCCALF有効化フラグ(sps_ccalf_enabled_flagなど)の値に基づいて、条件付きでシグナリングされる。

【0482】

ビットストリームは、CC-ALFを適用するときに復号において使用されるべき情報を信頼できる方法で提供しながら、サイズにおいて低減され得る。

【0483】

図12は、エンコーダ4000の実施形態を示す。エンコーダ4000は、ビデオを符号化する方法を実行するために、ここでは4001で示されている任意の処理回路を使用して実装され得る。具体的には、処理回路4001は、クロマ成分を精緻化するために、クロスコンポーネント適応ループフィルタCC-ALFを適用するように適合され得る。処理回路は、複数のCC-ALF関連構文要素を含むビットストリームを生成するようにさらに適合され得、複数のCC-ALF関連構文要素は、CC-ALF関連情報を示し、複数のCC-ALF関連構文要素は、シーケンスパラメータセット(SPS)レベル、ピクチャヘッダ、またはスライスヘッダのうちの任意の1つまたは複数においてシグナリングされる。

【0484】

この文脈における複数のCC-ALF関連構文要素は、第1の構文要素であって、クロスコンポーネント適応ループフィルタを含む適応ループフィルタ(ALF)がシーケンスレベルにおいて有効にされているかどうかを示し、SPSレベルにおいてシグナリングされる、第1の構文要素と、第2の構文要素であって、クロスコンポーネント適応ループフィルタがシーケンスレベルにおいて有効にされているかどうかを示し、SPSレベルにおいてシグナリングされる、第2の構文要素とを含む。

【0485】

エンコーダは、符号化されるべきビデオを受信するための受信機4002と、少なくとも

10

20

30

40

50

複数の構文要素を含むビットストリームを送信するための送信機4003とをさらに備え得る。

【0486】

図13は、デコーダ4100の実施形態を示す。デコーダ4100は、ビデオを復号する方法を実行するために任意の処理回路4101を使用して実装され得る。具体的には、処理回路4101は、ビットストリームからの複数のクロスコンポーネント適応ループフィルタCC-ALF関連構文要素を解析するように適合され得、複数のCC-ALF関連構文要素は、シーケンスパラメータセット(SPS)レベル、ピクチャヘッダ、またはスライスヘッダのうちの任意の1つまたは複数から取得される。

【0487】

具体的には、複数のCC-ALF関連構文要素は、第1の構文要素であって、クロスコンポーネント適応ループフィルタを含む適応ループフィルタ(ALF)がシーケンスレベルにおいて有効にされているかどうかを示し、SPSレベルにおいてシグナリングされる、第1の構文要素と、第2の構文要素であって、クロスコンポーネント適応ループフィルタがシーケンスレベルにおいて有効にされているかどうかを示し、SPSレベルにおいてシグナリングされる、第2の構文要素とを含む。

【0488】

処理回路は、複数のCC-ALF関連構文要素のうちの少なくとも1つを使用してCC-ALFプロセスを実行するようにさらに適合され得る。

【0489】

さらに、デコーダ4100は、ビットストリームを受信するための受信機4102を備え得る。それに加えて、デコーダは、復号されたビデオを、たとえば、ここには示されていない出力デバイスに出力するための送信機4103を備え得る。

【0490】

以下は、上記の実施形態に示すような符号化方法ならびに復号方法の適用と、それらを使用するシステムの説明である。

【0491】

図10は、コンテンツ配信サービスを実現するためのコンテンツ供給システム3100を示すブロック図である。このコンテンツ供給システム3100は、キャプチャデバイス3102と、端末デバイス3106とを含み、ディスプレイ3126をオプションで含む。キャプチャデバイス3102は、通信リンク3104を介して端末デバイス3106と通信する。通信リンクは、上記で説明した通信チャンネル13を含み得る。通信リンク3104は、限定はしないが、WIFI、イーサネット、ケーブル、ワイヤレス(3G/4G/5G)、USB、またはそれらの任意の種類

の組合せなどを含む。

【0492】

キャプチャデバイス3102は、データを生成し、上記の実施形態において示すような符号化方法によってデータを符号化し得る。代替的に、キャプチャデバイス3102は、データをストリーミングサーバ(図には示さず)に配信し得、サーバは、データを符号化し、符号化データを端末デバイス3106に送信する。キャプチャデバイス3102は、限定はしないが、カメラ、スマートフォンもしくはパッド、コンピュータもしくはラップトップ、ビデオ会議システム、PDA、車載デバイス、またはそれらの任意の種類の組合せなどを含む。たとえば、キャプチャデバイス3102は、上記で説明したようなソースデバイス12を含み得る。データがビデオを含む場合、キャプチャデバイス3102内に含まれるビデオエンコーダ20は、ビデオ符号化処理を実際に実行し得る。データがオーディオ(すなわち、音声)を含む場合、キャプチャデバイス3102内に含まれるオーディオエンコーダは、オーディオ符号化処理を実際に実行し得る。いくつかの実際的なシナリオについて、キャプチャデバイス3102は、符号化されたビデオおよびオーディオデータを、一緒に多重化することによって配信する。他の実際的なシナリオについて、たとえば、ビデオ会議システムにおいて、符号化オーディオデータおよび符号化ビデオデータは、多重化されない。キャプチャデバイス3102は、符号化オーディオデータおよび符号化ビデオデータを別々に端末デ

10

20

30

40

50

バイス3106に配信する。

【0493】

コンテンツ供給システム3100において、端末デバイス310は、符号化データを受信および再生する。端末デバイス3106は、上記の符号化データを復号することができるスマートフォンもしくはパッド3108、コンピュータもしくはラップトップ3110、ネットワークビデオレコーダ(NVR(network video recorder))/デジタルビデオレコーダ(DVR(digital video recorder))3112、TV3114、セットトップボックス(STB(set top box))3116、ビデオ会議システム3118、ビデオ監視システム3120、携帯情報端末(PDA(personal digital assistant))3122、車載デバイス3124、またはそれらの任意の組合せなどの、データ受信および再生能力を有するデバイスであり得る。たとえば、端末デバイス3106は、上記で説明したような宛先デバイス14を含み得る。符号化データがビデオを含む場合、端末デバイス内に含まれるビデオデコーダ30は、ビデオ復号を実行することを優先させる。符号化データがオーディオを有する場合、端末デバイス内に含まれるオーディオデコーダは、オーディオ復号処理を実行することを優先させる。

10

【0494】

ディスプレイを有する端末デバイス、たとえば、スマートフォンもしくはパッド3108、コンピュータもしくはラップトップ3110、ネットワークビデオレコーダ(NVR)/デジタルビデオレコーダ(DVR)3112、TV3114、携帯情報端末(PDA)3122、または車載デバイス3124について、端末デバイスは、復号されたデータをそのディスプレイに供給することができる。ディスプレイを備えない端末デバイス、たとえば、STB3116、ビデオ会議システム3118、またはビデオ監視システム3120について、復号されたデータを受信して表示するために、外部ディスプレイ3126がそこに接触させられる。

20

【0495】

このシステム内の各デバイスが符号化および復号を実行する場合、上記の実施形態において示すように、ピクチャ符号化デバイスまたはピクチャ復号デバイスが使用され得る。

【0496】

図11は、端末デバイス3106の一例の構造を示す図である。端末デバイス3106がキャプチャデバイス3102からストリームを受信した後、プロトコル進行ユニット3202は、ストリームの送信プロトコルを分析する。プロトコルは、限定はしないが、リアルタイムストリーミングプロトコル(RTSP(Real Time Streaming Protocol))、ハイパーテキスト転送プロトコル(HTTP(Hyper Text Transfer Protocol))、HTTPライブストリーミングプロトコル(HLS(HTTP Live streaming protocol))、MPEG-DASH、リアルタイム転送プロトコル(RTP(Real-time Transport protocol))、リアルタイムメッセージングプロトコル(RTMP(Real Time Messaging Protocol))、またはそれらの任意の種類の組合せなどを含む。

30

【0497】

プロトコル進行ユニット3202がストリームを処理した後、ストリームファイルが生成される。ファイルは、逆多重化ユニット3204に出力される。逆多重化ユニット3204は、多重化データを符号化オーディオデータと符号化ビデオデータとに分離することができる。上記で説明したように、いくつかの実際的なシナリオについて、たとえば、ビデオ会議システムにおいて、符号化オーディオデータおよび符号化ビデオデータは、多重化されない。このシナリオにおいて、符号化データは、逆多重化ユニット3204を介さずに、ビデオデコーダ3206およびオーディオデコーダ3208に送信される。

40

【0498】

逆多重化プロセスを介して、ビデオエレメンタリストリーム(ES(elementary stream))、オーディオES、およびオプションで字幕が生成される。上記の実施形態において説明したようなビデオデコーダ30を含むビデオデコーダ3206は、ビデオフレームを生成するために、上記の実施形態において示したような復号方法によってビデオESを復号し、このデータを同期ユニット3212に供給する。オーディオデコーダ3208は、オーディオフレームを生成するためにオーディオESを復号し、このデータを同期ユニット3212に供給する。

50

代替的に、ビデオフレームは、同期ユニット3212に供給する前に、バッファ(図Yには示さず)内に記憶し得る。同様に、オーディオフレームは、同期ユニット3212に供給する前に、バッファ(図Yには示さず)内に記憶し得る。

【0499】

同期ユニット3212は、ビデオフレームとオーディオフレームとを同期させ、ビデオ/オーディオをビデオ/オーディオディスプレイ3214に供給する。たとえば、同期ユニット3212は、ビデオおよびオーディオ情報の提示を同期させる。情報は、コーディングされたオーディオおよびビジュアルデータの提示に関するタイムスタンプと、データストリーム自体の配信に関するタイムスタンプとを使用して、構文においてコーディングし得る。

【0500】

字幕がストリーム内に含まれている場合、字幕デコーダ3210は、字幕を復号し、それをビデオフレームおよびオーディオフレームと同期させ、ビデオ/オーディオ/字幕をビデオ/オーディオ/字幕ディスプレイ3216に供給する。

【0501】

本発明は、上記のシステムに限定されず、上記の実施形態におけるピクチャ符号化デバイスまたはピクチャ復号デバイスのいずれかが、他のシステム、たとえば、自動車システムに組み込まれ得る。

【0502】

数学的演算子

本出願において使用される数学的演算子は、Cプログラミング言語において使用されるものと類似している。しかしながら、整数除算および算術シフト演算の結果は、より正確に定義され、べき乗および実数除算などの追加の演算が定義される。番号付けおよび計数の規則は、一般的に0から始まり、たとえば、「第1の」は、0番目に相当し、「第2の」は、1番目に相当する、などである。

【0503】

算術演算子

以下の算術演算子は、以下のように定義される。

+ 加算

- 減算(2引数演算子として)または否定(単項前置演算子として)

* 行列の乗算を含む乗算

XY べき乗。 x を y のべき乗に指定する。他の文脈において、このような表記は、べき乗としての解釈を意図しない上付き指定のために使用される。

/ 結果をゼロに向けて切り捨てる整数の除算。たとえば、 $7/4$ および $-7/-4$ は、1に切り捨てられ、 $-7/4$ および $7/-4$ は、-1に切り捨てられる。

$\div$ 切り捨てまたは丸めが意図されていない数学の方程式における除算を示すために使用される。

【数2】

$$\frac{x}{y}$$

切り捨てまたは丸めが意図されていない数学の方程式における除算を示すために使用される。

【数3】

10

20

30

40

50

$$\sum_{i=x}^y f(i)$$

yからyまでのすべての整数値を取るiによるf(i)の総和。

x%y モジュラス。x≥0かつy>0の整数xおよびyについてのみ定義される、xをyで割った余り。

【0504】

論理演算子

以下の論理演算子は、以下のように定義される。

x&y xおよびyのブール論理「and」

x|y xおよびyのブール論理「or」

! ブール論理「not」

x?y:z xが真であるか、または0に等しくない場合、yの値に評価され、そうでない場合、zの値に評価される。

【0505】

関係演算子

以下の関係演算子は、以下のように定義される。

> よりも大きい

>= 以上

< 未満

<= 以下

== 等しい

!= 等しくない

【0506】

値「na」（適用されない）が割り当てられた構文要素または変数に関係演算子が適用されると、値「na」は、構文要素または変数に関する個別の値として扱われる。値「na」は、他のどの値とも等しくないと見なされる。

【0507】

ビット演算子

以下のビット演算子は、以下のように定義される。

& ビット単位の「and」。整数の引数に対して演算する場合、整数値の2の補数表現に対して演算する。別の引数よりも少ないビットを含むバイナリ引数に対して演算する場合、短い方の引数は、0に等しいより多くの有効ビットを追加することによって拡張される。

| ビット単位の「or」。整数の引数に対して演算する場合、整数値の2の補数表現に対して演算する。別の引数よりも少ないビットを含むバイナリ引数に対して演算する場合、短い方の引数は、0に等しいより多くの有効ビットを追加することによって拡張される。

^ ビット単位の「排他的論理和」。整数の引数に対して演算する場合、整数値の2の補数表現に対して演算する。別の引数よりも少ないビットを含むバイナリ引数に対して演算する場合、短い方の引数は、0に等しいより多くの有効ビットを追加することによって拡張される。

x>>y yの2進数によるxの2の補数整数表現の算術右シフト。この関数は、yの非負の整数値に対してのみ定義される。右シフトの結果として最上位ビット(MSB(most significant bit))にシフトされたビットは、シフト演算前のxのMSBに等しい値を有する。

x<<y yの2進数によるxの2の補数整数表現の算術左シフト。この関数は、yの非負の整数値に対してのみ定義される。左シフトの結果として最下位ビット(LSB(least significant bit))にシフトされたビットは、0に等しい値を有する。

10

20

30

40

50

【0508】

代入演算子

以下の算術演算子は、以下のように定義される。

= 代入演算子

++ インクリメント、すなわち、x++は、x=x+1と等価であり、配列インデックスにおいて使用される場合、インクリメント演算の前の変数の値に評価される。

-- デクリメント、すなわち、x--は、x=x-1と等価であり、配列インデックスにおいて使用される場合、デクリメント演算の前の変数の値に評価される。

+= 指定された量によるインクリメントであり、すなわち、x+=3は、x=x+3と等価であり、x+=(-3)は、x=x+(-3)と等価である。

-= 指定された量によるデクリメントであり、すなわち、x-=3は、x=x-3と等価であり、x-=(-3)は、x=x-(-3)と等価である。

10

【0509】

範囲表記

以下の表記は、値の範囲を指定するために使用される。

x=y..z xは、yから開始してzまでの整数値を取り、x、yおよびzは、整数であり、zは、yよりも大きい。

【0510】

数学関数

以下の数学関数が定義される。

20

【0511】

【数4】

$$\text{Abs}(x) = \begin{cases} x & ; x \geq 0 \\ -x & ; x < 0 \end{cases}$$

【0512】

Asin(x) 三角関数の逆正弦関数であり、-1.0から1.0の範囲にある引数xに対して演算し、ラジアン単位において $-\pi/2$ から $\pi/2$ の範囲における出力値を有する

30

Atan(x) 三角関数の逆正接関数であり、引数xに対して演算し、ラジアン単位において $-\pi/2$ から $\pi/2$ の範囲における出力値を有する

【0513】

【数5】

$$\text{Atan2}(y, x) = \begin{cases} \text{Atan}\left(\frac{y}{x}\right) & ; x > 0 \\ \text{Atan}\left(\frac{y}{x}\right) + \pi & ; x < 0 \ \&\& \ y \geq 0 \\ \text{Atan}\left(\frac{y}{x}\right) - \pi & ; x < 0 \ \&\& \ y < 0 \\ +\frac{\pi}{2} & ; x = 0 \ \&\& \ y \geq 0 \\ -\frac{\pi}{2} & ; \text{それ以外の場合} \end{cases}$$

40

【0514】

50

Ceil(x) x以上の最小の整数。

Clip1_Y(x)=Clip3(0,(1 < <BitDepth_Y)-1,x)

Clip1_C(x)=Clip3(0,(1 < <BitDepth_C)-1,x)

【0 5 1 5】

【数 6】

$$\text{Clip3}(x, y, z) = \begin{cases} x & ; \quad z < x \\ y & ; \quad z > y \\ z & ; \quad \text{それ以外の場合} \end{cases}$$

10

【0 5 1 6】

Cos(x) ラジアン単位における引数xに対して演算する三角関数の余弦関数。

Floor(x) x以下の最大の整数。

【0 5 1 7】

【数 7】

$$\text{GetCurrMsb}(a, b, c, d) = \begin{cases} c+d & ; \quad b-a \geq d/2 \\ c-d & ; \quad a-b > d/2 \\ c & ; \quad \text{それ以外の場合} \end{cases}$$

20

【0 5 1 8】

Ln(x) xの自然対数(eを底とする対数、eは、自然対数の底定数2.718 281 828...)

Log2(x) xの2を底とする対数。

Log10(x) xの10を底とする対数。

【0 5 1 9】

【数 8】

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

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

40

【0 5 2 0】

Round(x)=Sign(x)*Floor(Abs(x)+0.5)

【0 5 2 1】

【数 9】

50

$$\text{Sign}(x) = \begin{cases} 1 & ; x > 0 \\ 0 & ; x == 0 \\ -1 & ; x < 0 \end{cases}$$

【0522】

10

$\text{Sin}(x)$ ラジアン単位における引数 x に対して演算する三角関数の正弦関数

【0523】

【数10】

$$\text{Sqrt}(x) = \sqrt{x}$$

【0524】

20

$\text{Swap}(x,y)=(y,x)$

$\text{Tan}(x)$ ラジアン単位における引数 x に対して演算する三角関数の正接関数

【0525】

演算の優先順位

式中の優先順位が括弧の使用によって明示的に示されていない場合、以下の規則が適用される。

- より高い優先順位の演算は、より低い優先順位の任意の演算の前に評価される。
- 同じ優先順位の演算は、左から右に順番に評価される。

【0526】

以下の表は、最高から最低までの演算の優先順位を指定し、表におけるより高い位置は、より高い優先順位を示す。

30

【0527】

Cプログラミング言語においても使用されるこれらの演算子について、この仕様において使用されている優先順位は、Cプログラミング言語において使用されているものと同じである。

【0528】

40

50

【表 4 1】

表:最高(表の上部)から最低(表の下部)までの演算優先順位

演算(オペランド x、y、および z を用いる)	
「x++」、 「x--」	
「!x」、 「-x」 (単項前置演算子として)	
x^y	
「x * y」、 「x / y」、 「x ÷ y」、 「 $\frac{x}{y}$ 」、 「x % y」	10
「x + y」、 「x - y」 (2 引数演算子として)、 「 $\sum_{i=x}^y f(i)$ 」	
「x << y」、 「x >> y」	
「x < y」、 「x <= y」、 「x > y」、 「x >= y」	
「x == y」、 「x != y」	
「x & y」	20
「x y」	
「x && y」	
「x y」	
「x ? y : z」	
「x..y」	
「x = y」、 「x += y」、 「x -= y」	

10

20

30

【0 5 2 9】

論理演算のテキスト説明

テキストにおいて、以下の形式、

if(condition 0)

statement 0

else if(condition 1)

statement 1

...

else /* informative remark on remaining condition */

statement n

において数学的に説明されるような論理演算の文は、以下の、

... as follows / ... the following applies:

- If condition 0, statement 0

- Otherwise, if condition 1, statement 1

- ...

- Otherwise (informative remark on remaining condition), statement n

のように記述され得る。

【0 5 3 0】

40

50

テキスト中の各「If ... Otherwise, if ... Otherwise, ...」文は、「... as follows」または「... the following applies」の直後に「If ...」が続く状態で導入される。「If ... Otherwise, if ... Otherwise, ...」の最後の条件は、常に「Otherwise, ...」である。インタリーブされた「If ... Otherwise, if ... Otherwise, ...」文は、「... as follows」または「... the following applies」を末尾の「Otherwise, ...」とマッチングさせることによって識別され得る。

【0531】

テキストにおいて、以下の形式、

```
if( condition 0a && condition 0b )
    statement 0
else if( condition 1a | | condition 1b )
    statement 1
...
else
    statement n
```

において数学的に記述される論理演算の文は、以下の、

... as follows / ... the following applies:

- If all of the following conditions are true, statement 0:
 - condition 0a
 - condition 0b
 - Otherwise, if one or more of the following conditions are true, statement 1:
 - condition 1a
 - condition 1b
 - ...
 - Otherwise, statement n
- のように記述され得る。

【0532】

テキストにおいて、以下の形式、

```
if( condition 0 )
    statement 0
if( condition 1 )
    statement 1
```

において数学的に記述される論理演算の文は、以下の、

When condition 0, statement 0

When condition 1, statement 1

のように記述され得る。

【0533】

本発明の実施形態について、主にビデオコーディングに基づいて説明してきたが、コーディングシステム10、エンコーダ20、およびデコーダ30(および対応してシステム10)の実施形態、ならびに本明細書で説明した他の実施形態は、静止ピクチャ処理またはコーディング、すなわち、ビデオコーディングのように任意の先行するまたは連続するピクチャとは独立した個々のピクチャの処理またはコーディングのためにも構成され得ることが留意されるべきである。一般に、ピクチャ処理コーディングが単一のピクチャ17に制限されている場合、インター予測ユニット244(エンコーダ)および344(デコーダ)のみが利用できない場合がある。ビデオエンコーダ20およびビデオデコーダ30のすべての他の機能(ツールまたは技術とも呼ばれる)、たとえば、残差計算204/304、変換206、量子化208、逆量子化210/310、(逆)変換212/312、パーティション分割262/362、イントラ予測254/354、および/またはループフィルタリング220、320、ならびにエントロピーコーディング270およびエントロピー復号204は、静止ピクチャ処理に対して等しく使用され得る

10

20

30

40

50

【0534】

たとえば、エンコーダ20およびデコーダ30の実施形態、ならびに、たとえば、エンコーダ20およびデコーダ30を参照して本明細書において説明した機能は、ハードウェア、ソフトウェア、ファームウェア、またはそれらの任意の組合せにおいて実装され得る。ソフトウェアにおいて実装される場合、機能は、1つまたは複数の命令またはコードとして、コンピュータ可読媒体上に記憶されるか、または通信媒体を介して送信され、ハードウェアベースの処理ユニットによって実行され得る。コンピュータ可読媒体は、データ記憶媒体などの有形の媒体に対応するコンピュータ可読記憶媒体、または、たとえば、通信プロトコルに従って、ある場所から別の場所へのコンピュータプログラムの転送を容易にする任意の媒体を含む通信媒体を含み得る。このように、コンピュータ可読媒体は、一般に、(1)非一時的である有形のコンピュータ可読記憶媒体、または(2)信号もしくは搬送波などの通信媒体に対応し得る。データ記憶媒体は、本開示において説明した技法の実装のための命令、コード、および/またはデータ構造を検索するために1つもしくは複数のコンピュータまたは1つもしくは複数のプロセッサによってアクセスされ得る任意の利用可能な媒体であり得る。コンピュータプログラム製品は、コンピュータ可読媒体を含み得る。

【0535】

例として、限定はしないが、そのようなコンピュータ可読記憶媒体は、RAM、ROM、EPROM、CD-ROMもしくは他の光ディスク記憶装置、磁気ディスク記憶装置もしくは他の磁気記憶装置、フラッシュメモリ、または所望のプログラムコードを命令もしくはデータ構造の形態において記憶するために使用され得、コンピュータによってアクセスされ得る任意の他の媒体を備え得る。また、任意の接続が、コンピュータ可読媒体と適切に呼ばれる。たとえば、命令が、同軸ケーブル、光ファイバケーブル、ツイストペア、デジタル加入者線(DSL(digital subscriber line))、または赤外線、無線、およびマイクロ波などのワイヤレス技術を使用してウェブサイト、サーバ、またはその他のリモートソースから送信される場合、同軸ケーブル、光ファイバケーブル、ツイストペア、DSL、または赤外線、無線、およびマイクロ波などのワイヤレス技術は、媒体の定義内に含まれる。しかしながら、コンピュータ可読記憶媒体およびデータ記憶媒体は、接続、搬送波、信号、またはその他の一時的媒体を含まず、その代わりに、非一時的な有形の記憶媒体に向けられることが理解されるべきである。本明細書において使用されるとき、ディスク(disk)およびディスク(disc)は、コンパクトディスク(CD(compact disc))、レーザーディスク(登録商標)、光ディスク、デジタル多用途ディスク(DVD(digital versatile disc))、フロッピー(登録商標)ディスク、およびBlu-ray(登録商標)ディスクを含み、ディスク(disk)は、通常、磁気的にデータを再生し、ディスク(disc)は、レーザーを用いて光学的にデータを再生する。上記のものの組合せも、コンピュータ可読媒体の範囲に含まれるべきである。

【0536】

命令は、1つまたは複数のデジタル信号プロセッサ(DSP)、汎用マイクロプロセッサ、特定用途向け集積回路(ASIC)、フィールドプログラマブルロジックアレイ(FPGA)、またはその他の等価な集積もしくはディスクリート論理回路などの1つまたは複数のプロセッサによって実行され得る。したがって、「プロセッサ」という用語は、本明細書において使用されるとき、上述の構造または本明細書において説明した技術の実装に適した任意の他の構造のいずれかを指し得る。それに加えて、いくつかの態様において、本明細書において説明した機能は、符号化および復号のために構成された専用のハードウェアおよび/もしくはソフトウェアモジュールにおいて提供されるか、または組み合わされたコーデックに組み込まれ得る。また、技法は、1つまたは複数の回路または論理要素においてすべて実装され得る。

【0537】

本開示の技法は、ワイヤレスハンドセット、集積回路(IC)、またはICのセット(たとえば、チップセット)を含む多種多様なデバイスまたは装置において実装され得る。様々な構成要素、モジュール、またはユニットについて、開示された技法を実行するように構成されたデバイスの機能的態様を強調するように本開示において説明したが、異なるハードウェア

10

20

30

40

50

アユニットによる実現を必ずしも必要としない。むしろ、上記で説明したように、様々なユニットが、コーデックハードウェアユニットにおいて組み合わせられるか、または適切なソフトウェアおよび/もしくはファームウェアと連携した、上記で説明した1つもしくは複数のプロセッサを含む相互運用性のあるハードウェアユニットの集合によって提供され得る。

【符号の説明】

【0538】

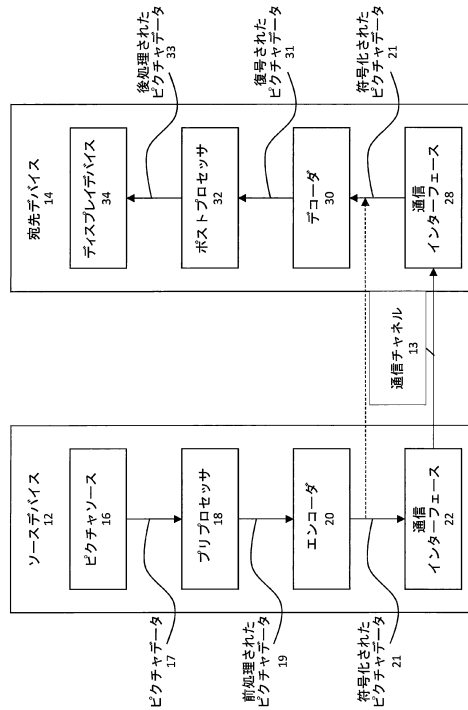
10	ビデオコーディングシステム	
12	ソースデバイス	
13	通信チャネル	10
14	宛先デバイス	
16	ピクチャソース	
17	ピクチャまたはピクチャデータ、生のピクチャまたは生のピクチャデータ	
18	プリプロセッサ、前処理ユニット	
19	前処理されたピクチャ、前処理されたピクチャデータ	
20	ビデオエンコーダ、非変換ベースのエンコーダ	
21	符号化ピクチャデータ、符号化ビットストリーム	
22	通信インターフェース、通信ユニット	
28	通信インターフェース、通信ユニット	
30	ビデオデコーダ、非変換ベースのデコーダ	20
31	復号されたピクチャ、復号されたピクチャデータ	
32	ポストプロセッサ、後処理ユニット	
33	後処理されたピクチャ、後処理されたピクチャデータ	
34	ディスプレイデバイス	
46	処理回路	
100	ビデオエンコーダ	
110	逆量子化ユニット	
201	入力、入力インターフェース	
203	ピクチャブロック、CTU	
204	残差計算ユニット	30
205	残差ブロック、残差	
206	変換処理ユニット	
207	変換係数	
208	量子化ユニット	
209	量子化された係数、量子化された変換係数、量子化された残差係数	
210	逆量子化ユニット	
211	脱量子化された係数、脱量子化された残差係数	
212	逆変換処理ユニット	
213	再構築された残差ブロック、脱量子化された係数、変換ブロック	
214	再構築ユニット、加算器、アナログ加算器	40
215	再構築されたブロック、フィルタリングされていない再構築されたブロック	
216	バッファ	
220	ループフィルタユニット、ループフィルタ	
221	フィルタリングされたブロック、フィルタリングされた再構築されたブロック	
230	復号ピクチャバッファ(DPB)	
244	インター予測ユニット	
254	イントラ予測ユニット	
260	モード選択ユニット	
262	パーティション分割ユニット	
265	予測ブロック、予測子、イントラ予測ブロック、インター予測ブロック	50

270	エントロピー符号化ユニット	
272	出力、出力インターフェース	
304	エントロピー復号ユニット	
309	量子化係数	
310	逆量子化ユニット	
311	脱量子化係数、変換係数	
312	逆変換処理ユニット	
313	再構築された残差ブロック、変換ブロック	
314	再構築ユニット、加算器、アナログ加算器	
315	再構築されたブロック	10
320	ループフィルタ、ループフィルタユニット	
321	復号されたビデオブロック、フィルタリングされたブロック	
330	復号ピクチャバッファ(DPB)	
331	復号されたピクチャ	
344	インター予測ユニット	
354	イントラ予測ユニット	
360	モード適用ユニット	
365	予測ブロック	
400	ビデオコーディングデバイス	
410	入口ポート、入力ポート	20
420	受信機ユニット(Rx)	
430	プロセッサ、論理ユニット、中央処理装置(CPU)	
440	送信機ユニット(Tx)	
450	出口ポート、出力ポート	
460	メモリ	
470	コーディングモジュール	
500	装置	
502	プロセッサ	
504	メモリ	
506	コードおよびデータ	30
508	オペレーティングシステム	
510	アプリケーションプログラム	
512	バス	
514	二次記憶装置	
518	ディスプレイ	
3100	コンテンツ供給システム	
3102	キャプチャデバイス	
3104	通信リンク	
3106	端末デバイス	
3108	スマートフォンまたはパッド	40
3110	コンピュータまたはラップトップ	
3112	ネットワークビデオレコーダ(NVR)/デジタルビデオレコーダ(DVR)	
3114	TV	
3116	セットトップボックス(STB)	
3118	ビデオ会議システム	
3120	ビデオ監視システム	
3122	携帯情報端末(PDA)	
3124	車載デバイス	
3126	ディスプレイ、外部ディスプレイ	
3202	プロトコル進行ユニット	50

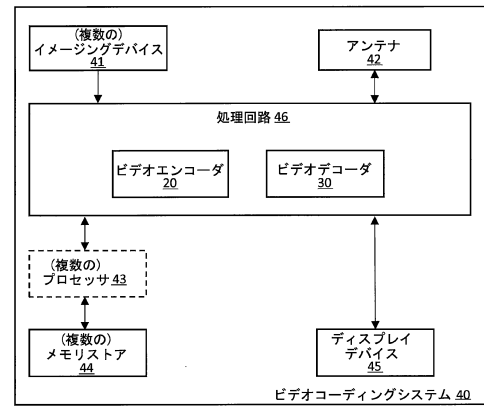
3204	逆多重化ユニット	
3206	ビデオデコーダ	
3208	オーディオデコーダ	
3210	字幕デコーダ	
3212	同期ユニット	
3214	ビデオ/オーディオディスプレイ	
3216	ビデオ/オーディオ/字幕ディスプレイ	
4000	エンコーダ	
4001	処理回路	
4002	受信機	10
4003	送信機	
4100	デコーダ	
4101	処理回路	
4102	受信機	
4103	送信機	
4200	方法	
4300	方法	
5000	データ構造、ビデオビットストリーム	
5005	VPS	
5010	SPS	20
5015	PPS	
5020	スライス	
5025	スライス	
5030	スライス	
5035	スライス	
5040	ヘッダ	
5045	データ	
5050	ブロック	
5055	ブロック	
5060	ブロック	30
7001	CCALFフィルタ情報	
8001	ピクチャヘッダエントリ	
8002	共通情報	

【図面】

【図 1 A】



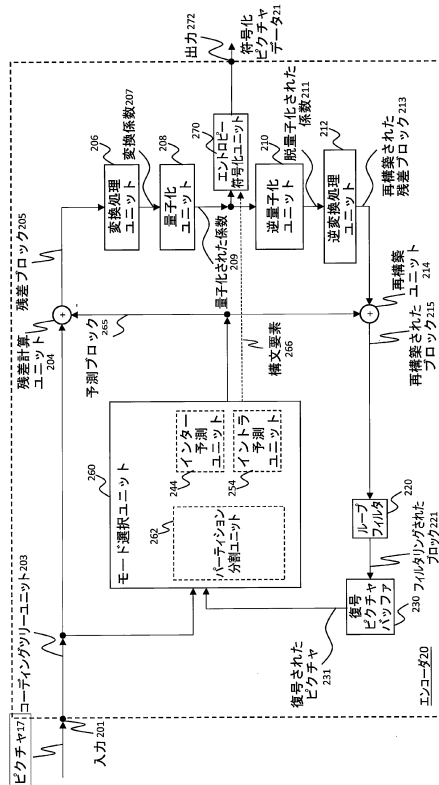
【図 1 B】



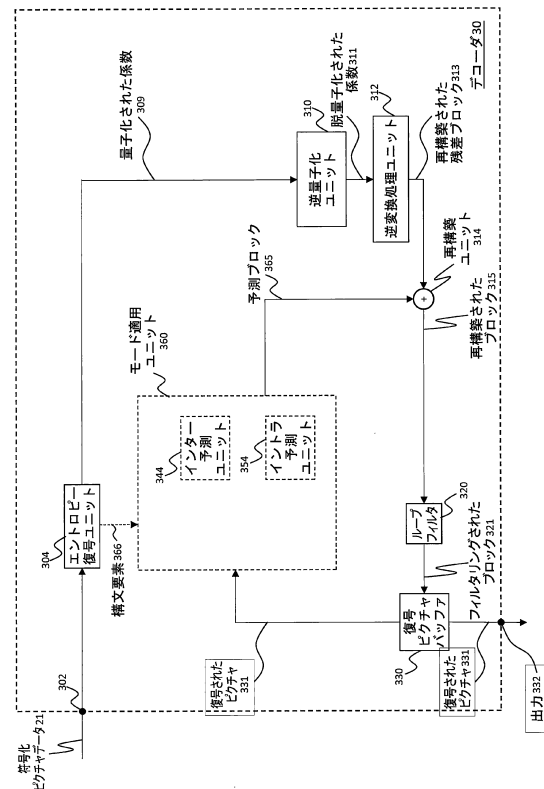
10

20

【図 2】



【図 3】

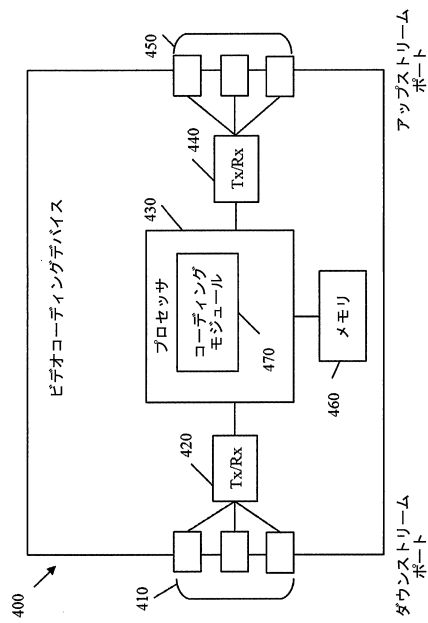


30

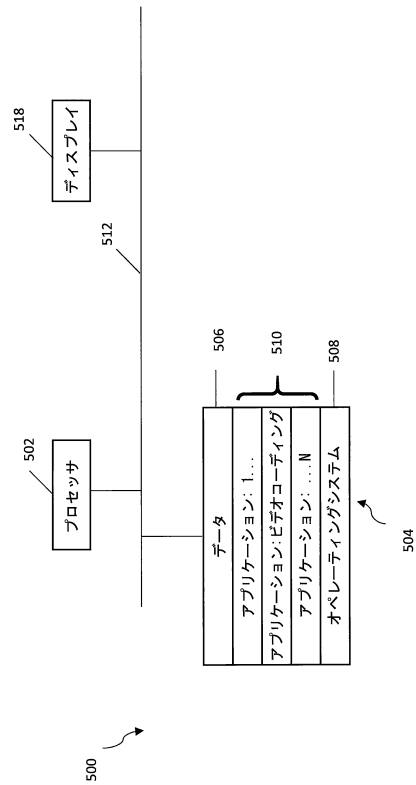
40

50

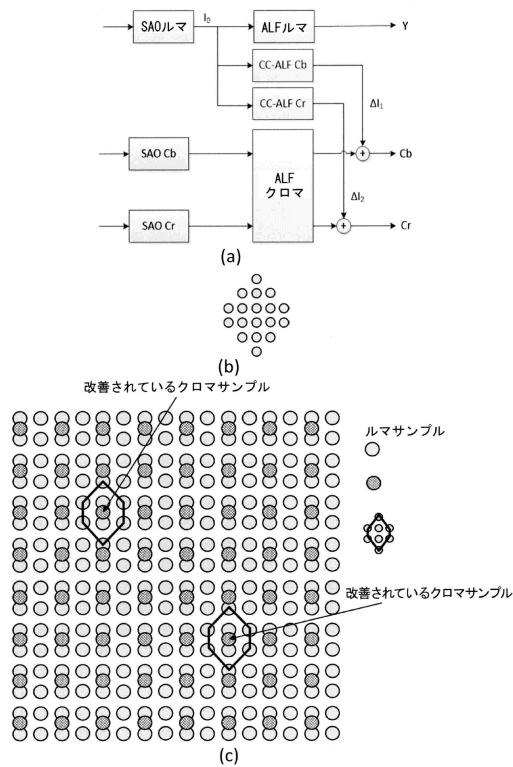
【図 4】



【図 5】



【図 6】



【図 7】



10

20

30

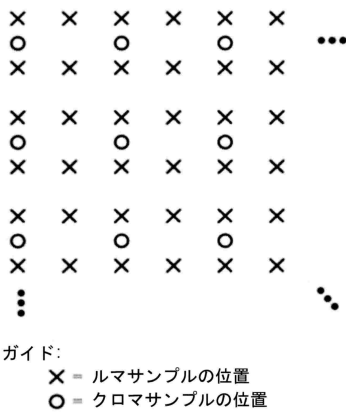
40

50

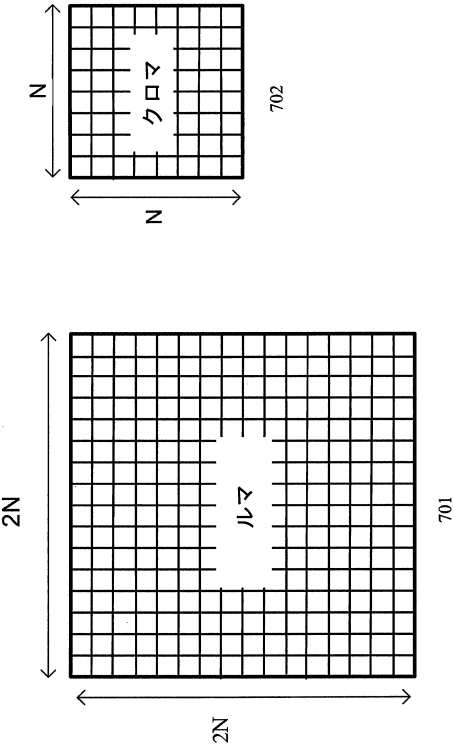
【図 8】



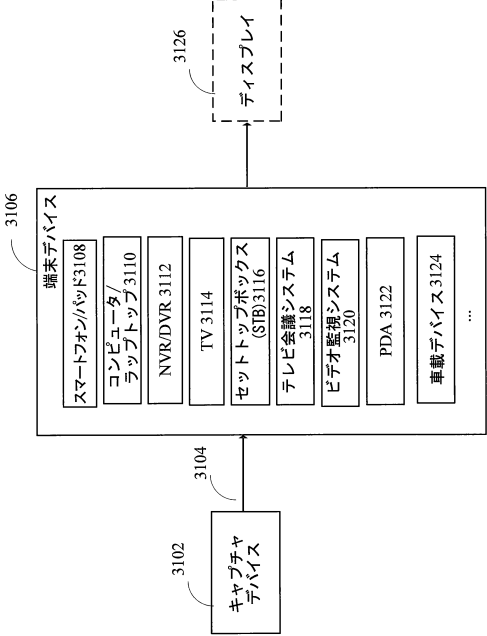
【図 9 A】



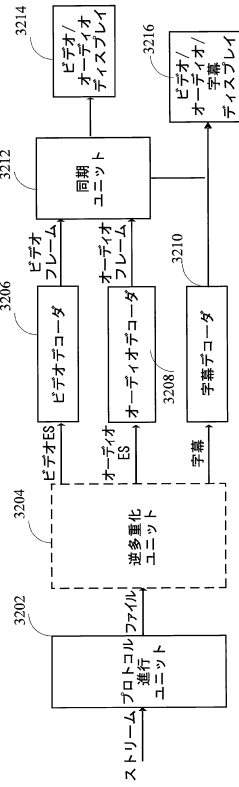
【図 9 B】



【図 10】



【図 1 1】



【図 1 2】

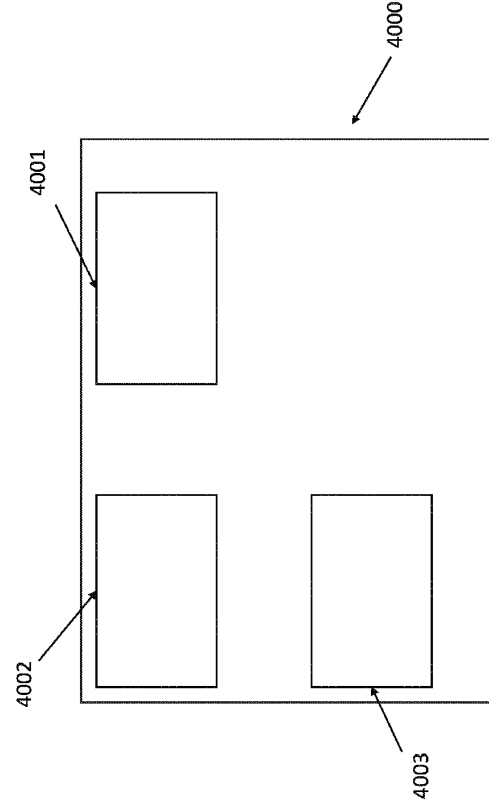


Fig. 12

【図 1 3】

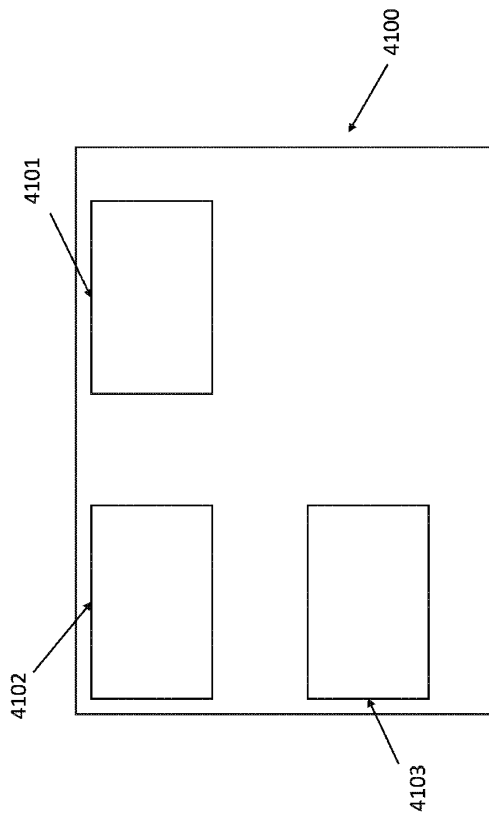
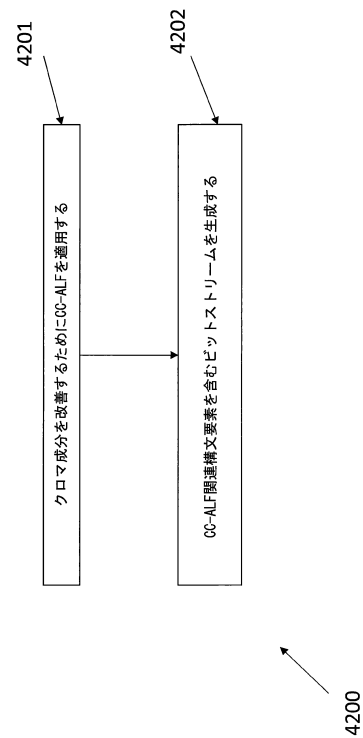


Fig. 13

【図 1 4】



10

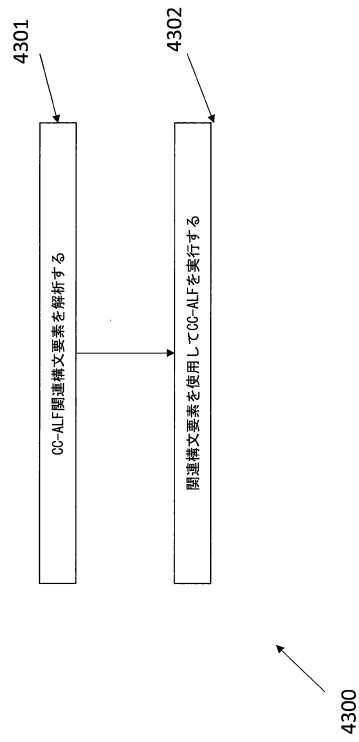
20

30

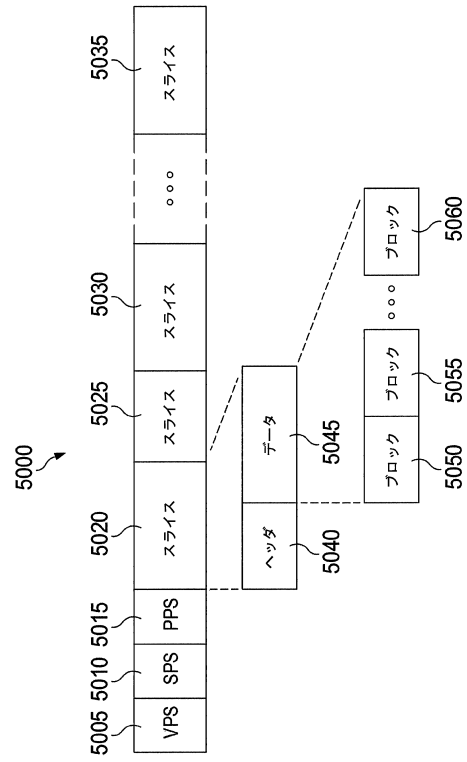
40

50

【図 15】



【図 16】



10

20

30

40

50

フロントページの続き

(33)優先権主張国・地域又は機関

米国(US)

クノロジーズ・デュッセルドルフ・ゲーエムベーハー

(72)発明者 セミフ・エセンリク

ドイツ・８０９９２・ミュンヘン・リースシュトラッセ・２５・ホアウェイ・テクノロジーズ・デュッセルドルフ・ゲーエムベーハー

(72)発明者 エレナ・アレクサンドロブナ・アルシナ

ドイツ・８０９９２・ミュンヘン・リースシュトラッセ・２５・ホアウェイ・テクノロジーズ・デュッセルドルフ・ゲーエムベーハー

(72)発明者 ハン・ガオ

ドイツ・８０９９２・ミュンヘン・リースシュトラッセ・２５・ホアウェイ・テクノロジーズ・デュッセルドルフ・ゲーエムベーハー

(72)発明者 ビャオ・ワン

ドイツ・８０９９２・ミュンヘン・リースシュトラッセ・２５・ホアウェイ・テクノロジーズ・デュッセルドルフ・ゲーエムベーハー

審査官 松元 伸次

(56)参考文献 特表２０２２－５４５８３８（ＪＰ，Ａ）

(58)調査した分野 (Int.Cl., DB名)

H04N 19／００－１９／９８