

(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号
特許第7578147号
(P7578147)

(45)発行日 令和6年11月6日(2024.11.6)

(24)登録日 令和6年10月28日(2024.10.28)

(51)国際特許分類

F I

H O 4 N 19/70 (2014.01)

H O 4 N 19/70

H O 4 N 19/85 (2014.01)

H O 4 N 19/85

請求項の数 23 (全34頁)

(21)出願番号	特願2022-579783(P2022-579783)	(73)特許権者	000002185
(86)(22)出願日	令和3年6月20日(2021.6.20)		ソニーグループ株式会社
(65)公表番号	特表2023-533680(P2023-533680		東京都港区港南1丁目7番1号
	A)	(74)代理人	100092093
(43)公表日	令和5年8月4日(2023.8.4)		弁理士 辻居 幸一
(86)国際出願番号	PCT/US2021/038163	(74)代理人	100109070
(87)国際公開番号	WO2021/262560		弁理士 須田 洋之
(87)国際公開日	令和3年12月30日(2021.12.30)	(74)代理人	100067013
審査請求日	令和4年12月22日(2022.12.22)		弁理士 大塚 文昭
(31)優先権主張番号	63/044,430	(74)代理人	
(32)優先日	令和2年6月26日(2020.6.26)		上杉 浩
(33)優先権主張国・地域又は機関		(74)代理人	100120525
	米国(US)		弁理士 近藤 直樹
(31)優先権主張番号	63/045,272	(72)発明者	グラジオッシ ダニエル
(32)優先日	令和2年6月29日(2020.6.29)		アメリカ合衆国 アリゾナ州 86001
	最終頁に続く		最終頁に続く

(54)【発明の名称】 V3C/V-PCCのための復号タイルハッシュSEIメッセージ

(57)【特許請求の範囲】

【請求項1】

ポイントクラウドコンテンツを符号化して、符号化されたポイントクラウドコンテンツを生成するステップと、
アトラスに対するハッシュ情報を生成するステップであって、前記ハッシュ情報を生成するステップは、前記アトラスのフレーム幅、前記アトラスのフレーム高さ及び変数値を含む前記アトラスのシンタックス要素の値を組み合わせ、シンタックス要素の組み合わせられた値にハッシュアルゴリズムを適用することを含み、前記ハッシュアルゴリズムは、複数のハッシュアルゴリズムを記憶するためのルックアップテーブルを用いて定義された変数に基づいて選択される、前記ハッシュ情報を生成するステップと、
前記ハッシュ情報を送信するステップと、

前記符号化されたポイントクラウドコンテンツを復号するステップと、
前記ハッシュ情報を使用して、前記復号されたコンテンツを検証するステップと、
を含むことを特徴とする方法。

【請求項2】

前記ポイントクラウドコンテンツを符号化するステップは、3Dから2Dへの投影に基づいてポイントクラウドを圧縮するV-PCC符号化実装を使用するステップを含み、前記ポイントクラウドコンテンツを符号化するステップは、3Dポイントクラウドデータを複数の2Dパッチにマッピングし、前記パッチを前記アトラス内に配置し、その後ビデオエンコーダで符号化するステップを含み、アトラス画像は、ポイントのジオメトリ、そ

それぞれのテクスチャ、及びどの位置をポイントクラウド再構成のために検討すべきであることを示す占有マップに対応することを特徴とする、請求項 1 に記載の方法。

【請求項 3】

前記ハッシュ情報は、前記アトラスの `block to patch` に対して送信されることを特徴とする、請求項 1 に記載の方法。

【請求項 4】

前記ハッシュ情報は、タイルに対して送信されることを特徴とする、請求項 1 に記載の方法。

【請求項 5】

前記ハッシュ情報は、タイルの `block to patch` に対して送信されることを特徴とする、請求項 1 に記載の方法。

【請求項 6】

前記ハッシュ情報は、補足強化情報（`SEI`）メッセージで送信されることを特徴とする、請求項 1 に記載の方法。

【請求項 7】

前記 `SEI` メッセージは、タイルベースの “`BlockToPatchMap [ ] [ ]`” ハッシュ値情報を含むことを特徴とする、請求項 6 に記載の方法。

【請求項 8】

前記ハッシュ情報は、`MD5`、`CRC` 及びチェックサムからなるグループから選択される実装を使用して生成されることを特徴とする、請求項 1 に記載の方法。

【請求項 9】

符号化されたポイントクラウドコンテンツを受け取り、
アトラスに対するハッシュ情報を受け取り、前記ハッシュ情報は、前記アトラスのフレーム幅、前記アトラスのフレーム高さ及び変数値を含む前記アトラスのシンタックス要素の組み合わせられた値を含み、シンタックス要素の組み合わせられた値にハッシュアルゴリズムを適用し、前記ハッシュアルゴリズムは、複数のハッシュアルゴリズムを記憶するためのルックアップテーブルを用いて定義された変数に基づいて選択され、

前記符号化されたポイントクラウドコンテンツを復号し、
前記ハッシュ情報を使用して、前記復号されたコンテンツを検証する、
ためのアプリケーションを記憶するための非一時的メモリと、
前記メモリに結合され、前記アプリケーションを処理するように構成されるプロセッサと、
を備えることを特徴とする装置。

【請求項 10】

前記ハッシュ情報は、前記アトラスの `block to patch` に対して送信されることを特徴とする、請求項 9 に記載の装置。

【請求項 11】

前記ハッシュ情報は、タイルに対して送信されることを特徴とする、請求項 9 に記載の装置。

【請求項 12】

前記ハッシュ情報は、タイルの `block to patch` に対して送信されることを特徴とする、請求項 9 に記載の装置。

【請求項 13】

前記ハッシュ情報は、補足強化情報（`SEI`）メッセージで送信されることを特徴とする、請求項 9 に記載の装置。

【請求項 14】

前記 `SEI` メッセージは、タイルベースの “`BlockToPatchMap [ ] [ ]`” ハッシュ値情報を含むことを特徴とする、請求項 13 に記載の装置。

【請求項 15】

前記ハッシュ情報は、`MD5`、`CRC` 及びチェックサムからなるグループから選択され

10

20

30

40

50

る実装を使用して生成されることを特徴とする、請求項 9 に記載の装置。

【請求項 16】

3次元コンテンツを取得するための1又は2以上のカメラと、
前記3次元コンテンツを符号化して、符号化されたポイントクラウドコンテンツを生成し、
アトラスに対するハッシュ情報を生成し、前記ハッシュ情報を生成することは、前記アトラスのフレーム幅、前記アトラスのフレーム高さ及び変数値を含む前記アトラスのシンタックス要素の値を組み合わせ、シンタックス要素の組み合わせられた値にハッシュアルゴリズムを適用することを含み、前記ハッシュアルゴリズムは、複数のハッシュアルゴリズムを記憶するためのルックアップテーブルを用いて定義された変数に基づいて選択され、前記ハッシュ情報を送信する、
ためのエンコーダと、
前記符号化されたポイントクラウドコンテンツを復号し、
前記ハッシュ情報を使用して、前記復号されたコンテンツを検証する、
ためのデコーダと、
を備えることを特徴とするシステム。

10

【請求項 17】

前記3次元コンテンツを符号化することは、3Dから2Dへの投影に基づいて3次元コンテンツを圧縮するV-PCC符号化実装を使用することを含み、前記3次元コンテンツを符号化することは、3Dポイントクラウドデータを複数の2Dパッチにマッピングし、
前記パッチを前記アトラス内に配置し、その後ビデオエンコーダで符号化することを含み、アトラス画像は、ポイントのジオメトリ、それぞれのテクスチャ、及びどの位置をポイントクラウド再構成のために検討すべきであることを示す占有マップに対応することを特徴とする、請求項 16 に記載のシステム。

20

【請求項 18】

前記ハッシュ情報は、前記アトラスの `block to patch` に対して送信されることを特徴とする、請求項 16 に記載のシステム。

【請求項 19】

前記ハッシュ情報は、タイルに対して送信されることを特徴とする、請求項 16 に記載のシステム。

30

【請求項 20】

前記ハッシュ情報は、タイルの `block to patch` に対して送信されることを特徴とする、請求項 16 に記載のシステム。

【請求項 21】

前記ハッシュ情報は、補足強化情報 (SEI) メッセージで送信されることを特徴とする、請求項 16 に記載のシステム。

【請求項 22】

前記SEIメッセージは、タイルベースの“`BlockToPatchMap [ ] [ ]`”ハッシュ値情報を含むことを特徴とする、請求項 21 に記載のシステム。

【請求項 23】

前記ハッシュ情報は、MD5、CRC及びチェックサムからなるグループから選択される実装を使用して生成されることを特徴とする、請求項 16 に記載のシステム。

40

【発明の詳細な説明】

【技術分野】

【0001】

〔関連出願との相互参照〕

本出願は、2020年7月1日に出願された「V3C/V-PCCのための復号タイルハッシュSEIメッセージ (DECODED TILE HASH SEI MESSAGE FOR V3C/V-PCC)」という名称の米国仮特許出願第63/046,749号、2020年6月29日に出願された「V3C/V-PCCのための復号タイルハッシュ

50

「V3C/V-PCCのための復号タイルハッシュSEIメッセージ」という名称の米国仮特許出願第63/045,272号、2020年6月26日に出版された「V3C/V-PCCのための復号タイルハッシュSEIメッセージ」という名称の米国仮特許出願第63/044,430号の米国特許法第119条に基づく優先権を主張するものであり、これらの文献はその全体が全ての目的で引用により本明細書に組み入れられる。

【0002】

本発明は、3次元グラフィックスに関する。具体的には、本発明は、3次元グラフィックスの符号化に関する。

【背景技術】

【0003】

近年、3Dから2Dへの投影に基づいてポイントクラウドを圧縮するための新規の方法が標準化されている。このV-PCC（ビデオベースのポイントクラウド圧縮）としても知られている方法は、3Dポイントクラウドデータを複数の2Dパッチにマッピングした後に、更にこれらのパッチをアトラス画像内に配置し、その後にビデオエンコーダで符号化する。アトラス画像は、ポイントのジオメトリ、それぞれのテクスチャ、及びどの位置をポイントクラウド再構成のために検討すべきであることを示す占有マップに対応する。

【0004】

2017年、MPEGは、ポイントクラウドの圧縮のための提案募集（CfP）を行った。複数の提案を評価した後に、MPEGは、（八分木及び同様の符号化法に基づく）3Dネイティブ符号化技術、又は3Dから2Dへの投影後に従来のビデオ符号化を行うもの、という2つの異なるポイントクラウド圧縮技術を現在検討している。動的3Dシーンの場合、MPEGは、パッチ表面モデリング、3Dから2D画像へのパッチの投影、及びHEVCなどのビデオエンコーダを使用した2D画像の符号化に基づく試験モデルソフトウェア（TMC2）を使用している。この方法は、ネイティブな3D符号化よりも効率的であり、容認可能な品質で競争的ビットレートを達成できることが証明されている。

【発明の概要】

【発明が解決しようとする課題】

【0005】

本明細書では、V3C/V-PCCアトラスフレームのための新しいハッシュ補足強化情報（SEI）メッセージについて説明する。このメッセージを使用して、各パッチと関連付けられるシンタックス要素について計算されるハッシュ値をシグナリングする。ハッシュSEIメッセージは、V3C/V-PCCタイルレベルのために、及び任意選択的にアトラスレベル適合性試験のために使用することができる。ハッシュを使用して、デコーダによって復号された値がエンコーダによって符号化された元の値と同じであることを確認することができる。

【課題を解決するための手段】

【0006】

一態様では、方法は、ポイントクラウドコンテンツを符号化して、符号化されたポイントクラウドコンテンツを生成するステップと、ハッシュ情報を生成して送信するステップと、前記符号化されたポイントクラウドコンテンツを復号するステップと、前記ハッシュ情報を使用して、前記復号されたコンテンツを検証するステップと、を含む。前記ポイントクラウドコンテンツを符号化するステップは、3Dから2Dへの投影に基づいてポイントクラウドを圧縮するV-PCC符号化実装を使用するステップを含み、前記ポイントクラウドコンテンツを符号化するステップは、3Dポイントクラウドデータを複数の2Dパッチにマッピングし、前記パッチをアトラス画像内に配置し、その後にビデオエンコーダで符号化するステップを含み、アトラス画像は、ポイントのジオメトリ、それぞれのテクスチャ、及びどの位置をポイントクラウド再構成のために検討すべきであることを示す占有マップに対応する。前記ハッシュ情報は、前記ポイントクラウドコンテンツのシンタックス要素及び/又は変数値を使用して生成される。前記ハッシュ情報は、アトラスに対して送信される。前記ハッシュ情報は、アトラスのblock to patchに対して送信さ

10

20

30

40

50

れる。前記ハッシュ情報は、タイルに対して送信される。前記ハッシュ情報は、タイルの `block to patch` に対して送信される。前記ハッシュ情報は、補足強化情報 (SEI) メッセージで送信される。前記 SEI メッセージは、タイルベースの “`BlockToPatchMap [ ] [ ]`” ハッシュ値情報を含む。前記ハッシュ情報は、MD5、CRC 及びチェックサムからなるグループから選択される実装を使用して生成される。
【0007】

別の態様では、装置は、符号化されたポイントクラウドコンテンツを受け取り、ハッシュ情報を受け取り、前記符号化されたポイントクラウドコンテンツを復号し、前記ハッシュ情報を使用して、前記復号されたコンテンツを検証する、ためのアプリケーションを記憶するための非一時的メモリと、前記メモリに結合され、前記アプリケーションを処理するように構成されるプロセッサと、を備える。前記ハッシュ情報は、前記ポイントクラウドコンテンツのシンタックス要素及び/又は変数値を使用して生成される。前記ハッシュ情報は、アトラスに対して送信される。前記ハッシュ情報は、アトラスの `block to patch` に対して送信される。前記ハッシュ情報は、タイルに対して送信される。前記ハッシュ情報は、タイルの `block to patch` に対して送信される。前記ハッシュ情報は、補足強化情報 (SEI) メッセージで送信される。前記 SEI メッセージは、タイルベースの “`BlockToPatchMap [ ] [ ]`” ハッシュ値情報を含む。前記ハッシュ情報は、MD5、CRC 及びチェックサムからなるグループから選択される実装を使用して生成される。

【0008】

別の態様では、システムは、3次元コンテンツを取得するための1又は2以上のカメラと、前記3次元コンテンツを符号化して、符号化されたポイントクラウドコンテンツを生成し、ハッシュ情報を生成して送信する、ためのエンコーダと、前記符号化されたポイントクラウドコンテンツを復号し、前記ハッシュ情報を使用して、前記復号されたコンテンツを検証する、ためのデコーダと、を備える。前記3次元コンテンツを符号化することは、3Dから2Dへの投影に基づいて3次元コンテンツを圧縮するV-PPC符号化実装を使用することを含み、前記3次元コンテンツを符号化することは、3Dポイントクラウドデータを複数の2Dパッチにマッピングし、前記パッチをアトラス画像内に配置し、その後ビデオエンコーダで符号化することを含み、アトラス画像は、ポイントのジオメトリ、それぞれのテクスチャ、及びどの位置をポイントクラウド再構成のために検討すべきであるかを示す占有マップに対応する。前記ハッシュ情報は、前記3次元コンテンツのシンタックス要素及び/又は変数値を使用して生成される。前記ハッシュ情報は、アトラスに対して送信される。前記ハッシュ情報は、アトラスの `block to patch` に対して送信される。前記ハッシュ情報は、タイルに対して送信される。前記ハッシュ情報は、タイルの `block to patch` に対して送信される。前記ハッシュ情報は、補足強化情報 (SEI) メッセージで送信される。前記 SEI メッセージは、タイルベースの “`BlockToPatchMap [ ] [ ]`” ハッシュ値情報を含む。前記ハッシュ情報は、MD5、CRC 及びチェックサムからなるグループから選択される実装を使用して生成される。

【図面の簡単な説明】

【0009】

【図1】いくつかの実施形態による、コンテンツを符号化及び復号する方法のフローチャートである。

【図2】いくつかの実施形態による、ハッシュを生成するフローチャートである。

【図3】いくつかの実施形態による、ハッシュを実装するように構成されるシステムの図である。

【図4】いくつかの実施形態による、復号タイルハッシュ SEI メッセージを実装するように構成される例示的なコンピュータ装置のブロック図である。

【発明を実施するための形態】

【0010】

本明細書では、V3C/V-PCCアトラスフレームのための新しいハッシュ補足強化情報（SEI）メッセージについて説明する。このメッセージを使用して、各パッチと関連付けられる全てのシンタックス要素について、タイル及びパッチのスキャン順序で計算されるハッシュ値をシグナリングする。ハッシュSEIメッセージは、V3C/V-PCCタイルレベルのために、及び任意選択的にアトラスレベル適合性試験のために使用することができることも断言する。

【0011】

ハッシュを使用して、デコーダによって復号された値がエンコーダによって符号化された元の値と同じであることを確認することができる。

【0012】

復号アトラス情報ハッシュSEIメッセージ

“BlockToPatchMap[][]”情報を正しく決定することができるデコーダが、ポイントクラウド再構成にとって重要な他の情報を適切に復号することができない場合があることが指摘されている。本明細書では、タイルベースの“BlockToPatchMap[][]”ハッシュ値情報を含むためのオプションを含むアトラス情報ハッシュSEIメッセージについて説明する。この組み合わせは、ブロックからボリュームへの情報の適切な復号、及び2Dから3Dへのパッチ変換に対応するボリュメトリックボックスの再構成を確実にする。

【0013】

復号アトラス情報ハッシュSEIメッセージのシンタックス

表1に、復号アトラス情報ハッシュSEIメッセージのシンタックス構造を示す。これはサフィックスSEIメッセージであり、タイプNAL_SUFFIX_SEIのNALユニットで搬送され、そのpayloadType値は21に設定される。復号アトラス情報ハッシュSEIメッセージの主な用途は、デバッグの目的のため、及びアトラスデコーダの適合性試験のためのものである。アトラスベース及びタイルベースの両方のハッシュ値計算のためのオプションが提供される。これは、タイル内の各パッチと関連付けられる全ての要素及びアトラス内の全てのタイルを経ることによって行われる。一例として、非EOM及びRAWパッチについて、変数Patch2dPosX[p]、Patch2dPosY[p]、Patch2dSizeX[p]、Patch2dSizeY[p]、Patch3dPosX[p]、Patch3dPosY[p]、Patch3dPosMinZ[p]、PatchOrientationIndex[p]、PatchLoDScaleX[p]、及びPatchLoDScaleY[p]を、予め特定された順序で、タイルの後にパッチが続く符号化順序で処理すべきである。EOM/RAWパッチが通常又は補助ストリームに関連するかどうかにも応じて、EOM/RAWパッチに対して同様の手法をとることもできる。アトラスベースのハッシュメッセージのチェックサム計算のための「フレームヘッダ」として使用される複数の高レベルの変数/シンタックス要素が含まれる。

【0014】

復号アトラス情報ハッシュSEIのシンタックス

10

20

30

40

50

【表 1】

decoded_atlas_info_hash(payloadSize) {	記述子	
daih_hash_type	u(8)	
daih_decoded_atlas_hash_present_flag	u(1)	
daih_decoded_atlas_b2p_hash_present_flag	u(1)	
daih_decoded_atlas_tiles_hash_present_flag	u(1)	10
daih_decoded_atlas_tiles_b2p_hash_present_flag	u(1)	
daih_reserved_4bits	u(4)	
if(decoded_atlas_hash_present_flag)		
if(daih_hash_type == 0)		
for(i = 0; i < 16; i++)		
daih_atlas_md5[i]	b(8)	20
else if(daih_hash_type == 1)		
daih_atlas_crc	u(16)	
else if(hash_type == 2)		
daih_atlas_checksum	u(32)	
if(decoded_atlas_b2p_hash_present_flag)		
if(daih_hash_type == 0)		30
for(i = 0; i < 16; i++)		
daih_atlas_b2p_md5[i]	b(8)	
else if(daih_hash_type == 1)		
daih_atlas_b2p_crc	u(16)	
else if(daih_hash_type == 2)		
daih_atlas_b2p_checksum	u(32)	40

```

    if( decoded_atlas_tiles_hash_present_flag ||
    decoded_atlas_tiles_b2p_hash_present_flag ) {

        daih_num_tiles_minus1                ue(v)

        daih_tile_id_len_minus1              ue(v)

        for( t = 0; t <= daih_num_tiles_minus1; t++ )
            daih_tile_id[ t ]                u(v)

        for( t = 0; t <= daih_num_tiles_minus1; t++ ) {

            j = TileIdToIndex[ daih_tile_id[ t ] ]

            if( decoded_atlas_tiles_hash_present_flag )

                if( daih_hash_type == 0 )

                    for( i = 0; i < 16; i++ )

                        daih_atlas_tiles_md5[ j ][ i ]

                        b(8)

                else if( daih_hash_type == 1 )

                    daih_atlas_tiles_crc[ j ]    u(16)

                else if( hash_type == 2 )

                    daih_atlas_tiles_checksum[ j ]    u(32)

            if( decoded_atlas_tiles_b2p_hash_present_flag )

                if( daih_hash_type == 0 )

                    for( i = 0; i < 16; i++ )

                        daih_atlas_tiles_b2p_md5[ j ][ i ]

                        b(8)

                else if( daih_hash_type == 1 )

                    daih_atlas_tiles_b2p_crc[ j ]    u(16)

```

10

20

30

40

50

```
else if( hash_type == 2 )  
  
    daih_atlas_tiles_b2p_checksum[j] = u(32)  
  
    }  
  
    }  
  
}
```

10

【0015】
復号アトラス情報SEIメッセージのセマンティクス
復号アトラス情報ハッシュSEIメッセージのフィールドのセマンティクスは、以下の通りである。

20

【0016】
まず、全てのシンタックス要素は、符号のない16ビットの数として記憶されると仮定する。アトラスレベルのハッシュ値の導出は、高レベルのシンタックス変数から開始して、以下のように示される。

【0017】

30

40

50

```

aLen = 0
atlasData[ aLen++ ] = asps_frame_width && 0XFF
atlasData[ aLen++ ] = (asps_frame_width >> 8) && 0XFF
atlasData[ aLen++ ] = asps_frame_height && 0XFF
atlasData[ aLen++ ] = (asps_frame_height >> 8) && 0XFF
atlasData[ aLen++ ] = AuxVideoWidth && 0XFF
atlasData[ aLen++ ] = (AuxVideoWidth >> 8) && 0XFF
atlasData[ aLen++ ] = oi_occupancy_nominal_2d_bitdepth_minus1 && 0XFF
atlasData[ aLen++ ] = (oi_occupancy_nominal_2d_bitdepth_minus1 >> 8) && 0XFF
atlasData[ aLen++ ] = oi_occupancy_MSB_align_flag && 0XFF
atlasData[ aLen++ ] = (oi_occupancy_MSB_align_flag >> 8) && 0XFF
atlasData[ aLen++ ] = oi_lossy_occupancy_map_compression_threshold && 0XFF
atlasData[ aLen++ ] = (oi_lossy_occupancy_map_compression_threshold >> 8) && 0XFF
atlasData[ aLen++ ] = vps_multiple_map_streams_present_flag && 0XFF
atlasData[ aLen++ ] = (vps_multiple_map_streams_present_flag >> 8) && 0XFF
atlasData[ aLen++ ] = vps_map_count_minus1 && 0XFF
atlasData[ aLen++ ] = (vps_map_count_minus1 >> 8) && 0XFF
for(i = 0; i < vps_map_count_minus1 + 1; i++) {
atlasData[ aLen++ ] = AttributeMapAbsoluteCodingEnabledFlag[ i ] && 0XFF
atlasData[ aLen++ ] = (AttributeMapAbsoluteCodingEnabledFlag[ i ] >> 8) && 0XFF
atlasData[ aLen++ ] = MapPredictorIndex[ i ] && 0XFF
atlasData[ aLen++ ] = (MapPredictorIndex[ i ] >> 8) && 0XFF
}
atlasData[ aLen++ ] = gi_geometry_nominal_2d_bitdepth_minus1 && 0XFF
atlasData[ aLen++ ] = (gi_geometry_nominal_2d_bitdepth_minus1 >> 8) && 0XFF
atlasData[ aLen++ ] = gi_geometry_3d_coordinates_bitdepth_minus1 && 0XFF
atlasData[ aLen++ ] = (gi_geometry_3d_coordinates_bitdepth_minus1 >> 8) && 0XFF
atlasData[ aLen++ ] = gi_geometry_MSB_align_flag && 0XFF
atlasData[ aLen++ ] = (gi_geometry_MSB_align_flag >> 8) && 0XFF
atlasData[ aLen++ ] = ai_attribute_count && 0XFF
atlasData[ aLen++ ] = (ai_attribute_count >> 8) && 0XFF
for(i = 0; i < ai_attribute_count; i++) {

```

10

20

30

40

50

```

atlasData[ aLen++ ] = ai_attribute_dimension_partitions_minus1[ i ] && 0xFF
atlasData[ aLen++ ] = (ai_attribute_dimension_partitions_minus1[ i ] >> 8) && 0xFF
for( j = 0; j < ai_attribute_dimension_partitions_minus1[ i ] + 1; j++ ) {
atlasData[ aLen++ ] = ai_attribute_partition_channels_minus1[ i ][ j ] && 0xFF
atlasData[ aLen++ ] = (ai_attribute_partition_channels_minus1[ i ][ j ] >> 8) && 0xFF
}

atlasData[ aLen++ ] = ai_attribute_nominal_2d_bitdepth_minus1[ i ] && 0xFF
atlasData[ aLen++ ] = (ai_attribute_nominal_2d_bitdepth_minus1[ i ] >> 8) && 0xFF
atlasData[ aLen++ ] = ai_attribute_MSB_align_flag[ i ] && 0xFF
atlasData[ aLen++ ] = (ai_attribute_MSB_align_flag[ i ] >> 8) && 0xFF
}

atlasData[ aLen++ ] = asps_geometry_3d_bitdepth_minus1 && 0xFF
atlasData[ aLen++ ] = (asps_geometry_3d_bitdepth_minus1 >> 8) && 0xFF
atlasData[ aLen++ ] = asps_geometry_2d_bitdepth_minus1 && 0xFF
atlasData[ aLen++ ] = (asps_geometry_2d_bitdepth_minus1 >> 8) && 0xFF
atlasData[ aLen++ ] = asps_map_count_minus1 && 0xFF
atlasData[ aLen++ ] = (asps_map_count_minus1 >> 8) && 0xFF
atlasData[ aLen++ ] = asps_pixel_deinterleaving_enabled_flag && 0xFF
atlasData[ aLen++ ] = (asps_pixel_deinterleaving_enabled_flag >> 8) && 0xFF
for( i = 0; i < asps_map_count_minus1 + 1; i++ ) {
atlasData[ aLen++ ] = asps_map_pixel_deinterleaving_flag[ i ] && 0xFF
atlasData[ aLen++ ] = (asps_map_pixel_deinterleaving_flag[ i ] >> 8) && 0xFF
}

atlasData[ aLen++ ] = asps_point_local_reconstruction_enabled_flag && 0xFF
atlasData[ aLen++ ] = (asps_point_local_reconstruction_enabled_flag >> 8) && 0xFF
for( i = 0; i < asps_map_count_minus1 + 1; i++ ) {
atlasData[ aLen++ ] = plri_point_local_reconstruction_map_flag[ i ] && 0xFF
atlasData[ aLen++ ] = (plri_point_local_reconstruction_map_flag[ i ] >> 8) && 0xFF
atlasData[ aLen++ ] = plri_number_of_modes_minus1[ i ] && 0xFF
atlasData[ aLen++ ] = (plri_number_of_modes_minus1[ i ] >> 8) && 0xFF
for( j = 0; j < plri_number_of_modes_minus1[ i ] + 1; j++ ) {
atlasData[ aLen++ ] = plri_interpolate_flag[ i ][ j ] && 0xFF
atlasData[ aLen++ ] = (plri_interpolate_flag[ i ][ j ] >> 8) && 0xFF
}
}

```

10

20

30

40

50

```

atlasData[ aLen++ ] = plri_filling_flag[ i ][ j ] && 0xFF
atlasData[ aLen++ ] = (plri_filling_flag[ i ][ j ] >> 8) && 0xFF
atlasData[ aLen++ ] = plri_minimum_depth[ i ][ j ] && 0xFF
atlasData[ aLen++ ] = (plri_minimum_depth[ i ][ j ] >> 8) && 0xFF
atlasData[ aLen++ ] = plri_neighbour_minus1[ i ][ j ] && 0xFF
atlasData[ aLen++ ] = (plri_neighbour_minus1[ i ][ j ] >> 8) && 0xFF
}
atlasData[ aLen++ ] = plri_block_threshold_per_patch_minus1[ i ] && 0xFF
atlasData[ aLen++ ] = (plri_block_threshold_per_patch_minus1[ i ] >> 8) && 0xFF
}
atlasData[ aLen++ ] = asps_eom_patch_enabled_flag && 0xFF
atlasData[ aLen++ ] = (asps_eom_patch_enabled_flag >> 8) && 0xFF
atlasData[ aLen++ ] = (asps_vpcc_surface_thickness_minus1 >> 8) && 0xFF
atlasData[ aLen++ ] = asps_vpcc_surface_thickness_minus1 && 0xFF
atlasData[ aLen++ ] = (asps_vpcc_surface_thickness_minus1 >> 8) && 0xFF
atlasData[ aLen++ ] = afps_output_flag_present_flag && 0xFF
atlasData[ aLen++ ] = (afps_output_flag_present_flag >> 8) && 0xFF

for( tileIdx = 0; tileIdx < afti_num_tiles_in_atlas_frame_minus1 + 1; tileIdx++ ) {
t = TileIndexToId[ tileIdx ]
    for( p = 0; p < AtduTotalNumberOfPatches[ t ]; p++ ) {

        if( ( atdu_patch_mode[ p ] == P_RAW ) || ( atdu_patch_mode[ p ] == I_RAW ) ) {
            if( rpdu_patch_in_auxiliary_video_flag[ t ][ p ] == 1 ) {
                atlasData[ aLen++ ] = AuxPatch2dPosX[ t ][ p ] & 0xFF

                atlasData[ aLen++ ] = (AuxPatch2dPosX[ t ][ p ] >> 8) & 0xFF
                atlasData[ aLen++ ] = AuxPatch2dPosY[ t ][ p ] & 0xFF

                atlasData[ aLen++ ] = (AuxPatch2dPosY[ t ][ p ] >> 8) & 0xFF
                atlasData[ aLen++ ] = AuxPatch2dSizeX[ t ][ p ] & 0xFF
                atlasData[ aLen++ ] = (AuxPatch2dSizeX[ t ][ p ] >> 8) &
0xFF

```

10

20

30

40

50

```

atlasData[ aLen++ ] = AuxPatch2dSizeY[ t ][ p ] & 0xFF
atlasData[ aLen++ ] = (AuxPatch2dSizeY[ t ][ p ] >> 8) &
0xFF

atlasData[ aLen++ ] = AuxPatch3dPosX[ t ][ p ] & 0xFF
atlasData[ aLen++ ] = (AuxPatch3dPosX[ t ][ p ] >> 8) &
0xFF

atlasData[ aLen++ ] = AuxPatch3dPosY[ t ][ p ] & 0xFF
atlasData[ aLen++ ] = (AuxPatch3dPosY[ t ][ p ] >> 8) &
0xFF

atlasData[ aLen++ ] = AuxPatch3dPosZ[ t ][ p ] & 0xFF
atlasData[ aLen++ ] = (AuxPatch3dPosZ[ t ][ p ] >> 8) &
0xFF

} else {
atlasData[ aLen++ ] = Patch2dPosX[ t ][ p ] & 0xFF
atlasData[ aLen++ ] = (Patch2dPosX[ t ][ p ] >> 8) &
0xFF

atlasData[ aLen++ ] = Patch2dPosY[ t ][ p ] & 0xFF
atlasData[ aLen++ ] = (Patch2dPosY[ t ][ p ] >> 8) &
0xFF

atlasData[ aLen++ ] = Patch2dSizeX[ t ][ p ] & 0xFF
atlasData[ aLen++ ] = (Patch2dSizeX[ t ][ p ] >> 8) &
0xFF

atlasData[ aLen++ ] = Patch2dSizeY[ t ][ p ] & 0xFF
atlasData[ aLen++ ] = (Patch2dSizeY[ t ][ p ] >> 8) &
0xFF

atlasData[ aLen++ ] = Patch3dPosX[ t ][ p ] & 0xFF
atlasData[ aLen++ ] = (Patch3dPosX[ t ][ p ] >> 8) &
0xFF

atlasData[ aLen++ ] = Patch3dPosY[ t ][ p ] & 0xFF
atlasData[ aLen++ ] = (Patch3dPosY[ t ][ p ] >> 8) &
0xFF

```

10

20

30

40

50

```

0xFF .
        atlasData[ aLen++ ] = Patch3dPosY[ t ][ p ] & 0xFF
        atlasData[ aLen++ ] = (Patch3dPosY[ t ][ p ] >> 8) &
0xFF
        atlasData[ aLen++ ] = Patch3dPosZ[ t ][ p ] & 0xFF
        atlasData[ aLen++ ] = (Patch3dPosZ[ t ][ p ] >> 8) &
0xFF
    }
    } else {

        if( ( atdu_patch_mode[ p ] == P_EOM ) || ( atdu_patch_mode[ p ] == I_EOM ) ) {
            atlasData[ aLen++ ] = EomPatch2dPosX[ t ][ p ] & 0xFF
            atlasData[ aLen++ ] = (EomPatch2dPosX[ t ][ p ] >> 8) &
0xFF
            atlasData[ aLen++ ] = EomPatch2dPosY[ t ][ p ] & 0xFF
            atlasData[ aLen++ ] = (EomPatch2dPosY[ t ][ p ] >> 8) &
0xFF
            atlasData[ aLen++ ] = EomPatch2dSizeX[ t ][ p ] &
0xFF
            atlasData[ aLen++ ] = (EomPatch2dSizeX[ t ][ p ] >> 8)
& 0xFF
            atlasData[ aLen++ ] = EomPatch2dSizeY[ t ][ p ] &
0xFF
            atlasData[ aLen++ ] = (EomPatch2dSizeY[ t ][ p ] >> 8)
& 0xFF
        } else {
            atlasData[ aLen++ ] = Patch2dPosX[ t ][ p ] & 0xFF
            atlasData[ aLen++ ] = (Patch2dPosX[ t ][ p ] >> 8) &
0xFF
            atlasData[ aLen++ ] = Patch2dPosY[ t ][ p ] & 0xFF
            atlasData[ aLen++ ] = (Patch2dPosY[ t ][ p ] >> 8) &
0xFF
            atlasData[ aLen++ ] = Patch2dSizeX[ t ][ p ] & 0xFF

```

10

20

30

40

50

```

                                atlasData[ aLen++ ] = (Patch2dSizeX[ t ][ p ] >> 8) &
0xFF
                                atlasData[ aLen++ ] = Patch2dSizeY[ t ][ p ] & 0xFF
                                atlasData[ aLen++ ] = (Patch2dSizeY[ t ][ p ] >> 8) &
0xFF
                                atlasData[ aLen++ ] = Patch3dPosX[ t ][ p ] & 0xFF
                                atlasData[ aLen++ ] = (Patch3dPosX[ t ][ p ] >> 8) &
0xFF
                                atlasData[ aLen++ ] = Patch3dPosY[ t ][ p ] & 0xFF
                                atlasData[ aLen++ ] = (Patch3dPosY[ t ][ p ] >> 8) &
0xFF
                                atlasData[ aLen++ ] = Patch3dPosMinZ[ t ][ p ] & 0xFF
                                atlasData[ aLen++ ] = (Patch3dPosMinZ[ t ][ p ] >> 8) &
0xFF
                                atlasData[ aLen++ ] = PatchOrientationIndex[ t ][ p ] &
0xFF
                                atlasData[ aLen++ ] = (PatchOrientationIndex[ t ][ p ] >> 8) & 0xFF
                                atlasData[ aLen++ ] = PatchLoDScaleX[ t ][ p ] & 0xFF
                                atlasData[ aLen++ ] = (PatchLoDScaleX[ t ][ p ] >> 8) &
0xFF
                                atlasData[ aLen++ ] = PatchLoDScaleY[ t ][ p ] & 0xFF
                                atlasData[ aLen++ ] = (PatchLoDScaleY[ t ][ p ] >> 8) &
0xFF
                                }
                                }
                                }
                                }

```

【 0 0 1 8 】

同様に、アトラスベースの `BlockToPatch [ ] [ ]` については、以下の通りである。

```

b2paLen = 0
for( tileIdx = 0; tileIdx < afti_num_tiles_in_atlas_frame_minus1 + 1; tileIdx++ ) {
    t = TileIndexToId[ tileIdx ]
    BlockToPatchMapWidth = Ceil(TileGroupWidth[ t ]/PatchPackingBlockSize)
    BlockToPatchMapHeight = Ceil(TileGroupHeight[ t ]/PatchPackingBlockSize)
    for( y = 0; y < BlockToPatchMapHeight; y++ ) {
        for( x = 0; x < BlockToPatchMapWidth ; x++ ) {
            atlasB2pData[ b2paLen++ ] = BlockToPatchMap[ y ][ x ] & 0xFF
            atlasB2pData[ b2paLen++ ]
            = (BlockToPatchMap[ y ][ x ] >> 8) & 0xFF
        }
    }
}

```

【 0 0 1 9 】

タイルベースのハッシュ値の導出は、以下のように示される。

30

40

50

```

for( tileIdx = 0; tileIdx < afti_num_tiles_in_atlas_frame_minus1 + 1; tileIdx++ ) {
    t = TileIndexToId[ tileIdx ]
    tLen = 0
    for( p = 0; p < AtduTotalNumberOfPatches[ t ]; p++ ) {

        if( ( atdu_patch_mode[ p ] == P_RAW ) || ( atdu_patch_mode[ p ] == I_RAW ) ) {
            if( rpdu_patch_in_auxiliary_video_flag[ t ][ p ] == 1 ) {
                tileData[ t ][ tLen++ ] = AuxPatch2dPosX[ t ][ p ] & 0xFF

                tileData[ t ][ tLen++ ] = (AuxPatch2dPosX[ t ][ p ] >> 8) & 0xFF

                tileData[ t ][ tLen++ ] = AuxPatch2dPosY[ t ][ p ] & 0xFF

                tileData[ t ][ tLen++ ] = (AuxPatch2dPosY[ t ][ p ] >> 8) & 0xFF
                tileData[ t ][ tLen++ ] = AuxPatch2dSizeX[ t ][ p ] &
0xFF
                tileData[ t ][ tLen++ ] = (AuxPatch2dSizeX[ t ][ p ] >> 8)
& 0xFF
                tileData[ t ][ tLen++ ] = AuxPatch2dSizeY[ t ][ p ] &
0xFF
                tileData[ t ][ tLen++ ] = (AuxPatch2dSizeY[ t ][ p ] >> 8)
& 0xFF
                tileData[ t ][ tLen++ ] = AuxPatch3dPosX[ t ][ p ] &
0xFF
                tileData[ t ][ tLen++ ] = (AuxPatch3dPosX[ t ][ p ] >> 8)
& 0xFF
                tileData[ t ][ tLen++ ] = AuxPatch3dPosY[ t ][ p ] &
0xFF
                tileData[ t ][ tLen++ ] = (AuxPatch3dPosY[ t ][ p ] >> 8)
& 0xFF
                tileData[ t ][ tLen++ ] = AuxPatch3dPosY[ t ][ p ] &
0xFF
            }
        }
    }
}

```

10

20

30

40

50

```

0xFF
tileData[ t ][ tLen++ ] = (AuxPatch3dPosY[ t ][ p ] >> 8)
& 0xFF
tileData[ t ][ tLen++ ] = AuxPatch3dPosZ[ t ][ p ] &
0xFF
tileData[ t ][ tLen++ ] = (AuxPatch3dPosZ[ t ][ p ] >> 8)
& 0xFF
} else {
tileData[ t ][ tLen++ ] = Patch2dPosX[ t ][ p ] & 0xFF
tileData[ t ][ tLen++ ] = (Patch2dPosX[ t ][ p ] >> 8) &
0xFF
tileData[ t ][ tLen++ ] = Patch2dPosY[ t ][ p ] & 0xFF
tileData[ t ][ tLen++ ] = (Patch2dPosY[ t ][ p ] >> 8) &
0xFF
tileData[ t ][ tLen++ ] = Patch2dSizeX[ t ][ p ] & 0xFF
tileData[ t ][ tLen++ ] = (Patch2dSizeX[ t ][ p ] >> 8) &
0xFF
tileData[ t ][ tLen++ ] = Patch2dSizeY[ t ][ p ] & 0xFF
tileData[ t ][ tLen++ ] = (Patch2dSizeY[ t ][ p ] >> 8) &
0xFF
tileData[ t ][ tLen++ ] = Patch3dPosX[ t ][ p ] & 0xFF
tileData[ t ][ tLen++ ] = (Patch3dPosX[ t ][ p ] >> 8) &
0xFF
tileData[ t ][ tLen++ ] = Patch3dPosY[ t ][ p ] & 0xFF
tileData[ t ][ tLen++ ] = (Patch3dPosY[ t ][ p ] >> 8) &
0xFF
tileData[ t ][ tLen++ ] = Patch3dPosY[ t ][ p ] & 0xFF
tileData[ t ][ tLen++ ] = (Patch3dPosY[ t ][ p ] >> 8) &
0xFF
tileData[ t ][ tLen++ ] = Patch3dPosZ[ t ][ p ] & 0xFF
tileData[ t ][ tLen++ ] = (Patch3dPosZ[ t ][ p ] >> 8) &
0xFF
}

```

10

20

30

40

50

```

    } else {

        if( ( atdu_patch_mode[ p ] == P_EOM ) || ( atdu_patch_mode[ p ] == I_EOM ) ) {
            tileData[ t ][ tLen++ ] = EomPatch2dPosX[ t ][ p ] &
0xFF
            tileData[ t ][ tLen++ ] = (EomPatch2dPosX[ t ][ p ] >>
8) & 0xFF
10
            tileData[ t ][ tLen++ ] = EomPatch2dPosY[ t ][ p ] &
0xFF
            tileData[ t ][ tLen++ ] = (EomPatch2dPosY[ t ][ p ] >>
8) & 0xFF
            tileData[ t ][ tLen++ ] = EomPatch2dSizeX[ t ][ p ] &
0xFF
            tileData[ t ][ tLen++ ] = (EomPatch2dSizeX[ t ][ p ] >>
8) & 0xFF
            tileData[ t ][ tLen++ ] = EomPatch2dSizeY[ t ][ p ] &
20
            tileData[ t ][ tLen++ ] = (EomPatch2dSizeY[ t ][ p ] >>
8) & 0xFF

        } else {
            tileData[ t ][ tLen++ ] = Patch2dPosX[ t ][ p ] & 0xFF
            tileData[ t ][ tLen++ ] = (Patch2dPosX[ t ][ p ] >> 8) &
0xFF
            tileData[ t ][ tLen++ ] = Patch2dPosY[ t ][ p ] & 0xFF
            tileData[ t ][ tLen++ ] = (Patch2dPosY[ t ][ p ] >> 8) &
30
            tileData[ t ][ tLen++ ] = Patch2dSizeX[ t ][ p ] & 0xFF
            tileData[ t ][ tLen++ ] = (Patch2dSizeX[ t ][ p ] >> 8) &
0xFF
            tileData[ t ][ tLen++ ] = Patch2dSizeY[ t ][ p ] & 0xFF
            tileData[ t ][ tLen++ ] = (Patch2dSizeY[ t ][ p ] >> 8) &
0xFF
            tileData[ t ][ tLen++ ] = Patch3dPosX[ t ][ p ] & 0xFF
40

```

```

tileData[t][tLen++] = (Patch3dPosX[t][p] >> 8) &
0xFF

tileData[t][tLen++] = Patch3dPosY[t][p] & 0xFF
tileData[t][tLen++] = (Patch3dPosY[t][p] >> 8) &
0xFF

tileData[t][tLen++] = Patch3dPosMinZ [t][p] &
0xFF
10

tileData[t][tLen++] = (Patch3dPosMinZ [t][p] >>
8) & 0xFF

tileData[t][tLen++] = PatchOrientationIndex [t][p]
& 0xFF

tileData[t][tLen++] = (PatchOrientationIndex [t][p]
>> 8) & 0xFF

tileData[t][tLen++] = PatchLoDScaleX [t][p] &
0xFF
20

tileData[t][tLen++] = (PatchLoDScaleX [t][p] >>
8) & 0xFF

tileData[t][tLen++] = PatchLoDScaleY [t][p] &
0xFF

tileData[t][tLen++] = (PatchLoDScaleY [t][p] >>
8) & 0xFF
30

    }
}
}
dataLen[t] = tLen
}

```

【 0 0 2 0 】

タイルベースの `BlockToPatch [ ] [ ]` については、`atlasB2pD`
`ata [ ]` アレイが、以下のように生成される。

40

50

```

for( tileIdx = 0; tileIdx < afti_num_tiles_in_atlas_frame_minus1 + 1; tileIdx++ ) {
    t = TileIndexToId[ tileIdx ]
    b2ptLen=0
        BlockToPatchMapWidth = Ceil(TileGroupWidth[ t ]/PatchPackingBlockSize)
        BlockToPatchMapHeight = Ceil(TileGroupHeight[ t ]/PatchPackingBlockSize)
        for( y = 0; y < BlockToPatchMapHeight; y++ ) {
            for( x = 0; x < BlockToPatchMapWidth ; x++ ) {
                atlasB2pData[ b2paLen++ ] = tileB2pData[ b2ptLen++ ] = BlockToPatchMap[ y ][ x ] & 0xFF
                atlasB2pData[ b2paLen++ ] = tileB2pData[ b2ptLen++ ] = (BlockToPatchMap[ y ][ x ] >> 8) & 0xFF
            }
        }
        dataB2pLen[ t ] = b2ptLen
    }
}

```

10

20

【0021】

`daih__hash__type` は、以下の表 2 に従ってチェックサムを計算するために使用される方法を示す。表に列挙されていない `daih__hash__type` の値は、ITU-T | ISO/IEC によって今後の使用のために留保され、この仕様のこのバージョンに準拠するビットストリーム内に存在しない。デコーダは、留保された `hash__type` の値を含む復号ピクチャハッシュ SEI メッセージを無視する。

30

【0022】

アトラスハッシュタイプ

【表 2】

hash_type	方法
0	MD5
1	CRC
2	チェックサム

40

【0023】

`daih__decoded__atlas__hash__present__flag` が 1 に等しい場合には、`daih__atlas__md5 [ i ]`、`daih__atlas__crc` 又は `daih__atlas__checksum` シンタックス要素が存在することを特定する。`daih__decoded__atlas__hash__present__flag` が 0 に等しい場合には、`daih__atlas__md5 [ i ]`、`daih__atlas__crc` 又は `daih__atlas__checksum` シンタックス要素が存在しないことを特定する。

【0024】

50

`daih_decoded_atlas_b2p_hash_present_flag`が1に等しい場合には、`daih_atlas_b2p_md5[i]`、`daih_atlas_b2p_crc`又は`daih_atlas_b2p_checksum`シンタックス要素が存在することを特定する。`daih_decoded_atlas_b2p_hash_present_flag`が0に等しい場合には、`daih_atlas_b2p_md5[i]`、`daih_atlas_b2p_crc`又は`daih_atlas_b2p_checksum`が存在しないことを特定する。

【0025】

`daih_decoded_atlas_tiles_hash_present_flag`が1に等しい場合には、`daih_atlas_tiles_md5[i]`、`daih_atlas_tiles_crc`又は`daih_atlas_tiles_checksum`シンタックス要素が存在することを特定する。`daih_decoded_atlas_tiles_hash_present_flag`が0に等しい場合には、`daih_atlas_tiles_md5[i]`、`daih_atlas_tiles_crc`又は`daih_atlas_tiles_checksum`シンタックス要素が存在しないことを特定する。

10

【0026】

`daih_atlas_tiles_b2p_hash_present_flag`が1に等しい場合には、`daih_atlas_tiles_b2p_md5[i]`、`daih_atlas_tiles_b2p_crc`又は`daih_atlas_tiles_b2p_checksum`シンタックス要素が存在することを特定する。`daih_atlas_tiles_b2p_hash_present_flag`が0に等しい場合には、`daih_atlas_tiles_b2p_md5[i]`、`daih_atlas_tiles_b2p_crc`又は`daih_atlas_tiles_b2p_checksum`が存在しないことを特定する。

20

【0027】

`daih_atlas_md5[i]`は、`vuh_atlas_id`と関連付けられる復号アトラスの16バイトのMD5ハッシュである。`daih_atlas_md5[i]`の値は、IETF RFC 1321において定められるMD5関数を使用して以下のよう
に得られる`digestVal`の値に等しい。

30

`MD5Init(context)`

`MD5Update(context, atlasData, aLen)`

`MD5Final(digestVal, context)`

【0028】

`daih_atlas_crc`は、`vuh_atlas_id`と関連付けられる復号アトラスの周期冗長検査(CRC)である。`daih_atlas_crc`の値は、Rec. ITU-T H. 271において定められるCRC仕様を使用して以下のよう
に得られる`crcVal`の値に等しい。

40

```

crc = 0xFFFF
atlasData[ aLen ] = 0
atlasData[ aLen + 1 ] = 0
for( bitIdx = 0; bitIdx < ( aLen + 2 ) * 8; bitIdx++ ) {
    dataByte = atlasData[ bitIdx >> 3 ]
    crcMsb = ( crc >> 15 ) & 1
    bitVal = ( dataByte >> ( 7 - ( bitIdx & 7 ) ) ) & 1
    crc = ( ( ( crc << 1 ) + bitVal ) & 0xFFFF ) ^ ( crcMsb * 0x1021 )
}
crcVal = crc

```

10

【 0 0 2 9 】

`d a i h _ a t l a s _ c h e c k s u m`は、`v u h _ a t l a s _ i d`と関連付けられる復号アトラスのチェックサムである。`d a i h _ a t l a s _ c h e c k s u m`の値は、以下のように得られる`c h e c k s u m V a l`の値に等しい。

20

```

checksum = 0
for( i = 0; i < aLen; i++ ) {
    xorMask = ( i & 0xFF ) ^ ( i >> 8 )
    checksum = ( checksum + ( atlasData[ i ] & 0xFF ) ^ xorMask ) &
0xFFFFFFFF
    checksum = ( checksum + ( atlasData[ i ] >> 8 ) ^ xorMask ) &
0xFFFFFFFF
}

```

30

【 0 0 3 0 】

`d a i h _ a t l a s _ b 2 p _ m d 5 [ i ]`は、`v u h _ a t l a s _ i d`と関連付けられるアトラスの`B l o c k T o P a t c h [ ] [ ]`の16バイトのMD5ハッシュである。`d a i h _ a t l a s _ b 2 p _ m d 5 [ i ]`の値は、I E T F R F C 1 3 2 1において定められるMD5関数を使用して以下のように得られる`d i g e s t V a l`の値に等しい。

40

```

MD5Init( context )
MD5Update( context, atlasB2pData, b2paLen )
MD5Final( digestVal, context )

```

【 0 0 3 1 】

50

`daih__atlas__b2p__crc`は、`vuh__atlas__id`と関連付けられる復号アトラスの周期冗長検査（CRC）である。`daih__atlas__b2p__crc`の値は、Rec. ITU-T H. 271において定められるCRC仕様を使用して以下のように得られる`crcVal`の値に等しい。

```

crc = 0xFFFF
atlasB2pData[ b2paLen ] = 0
atlasB2pData[ b2paLen + 1 ] = 0
for( bitIdx = 0; bitIdx < ( b2paLen + 2 ) * 8; bitIdx++ ) {
    dataByte = atlasB2pData[ bitIdx >> 3 ]
    crcMsb = ( crc >> 15 ) & 1
    bitVal = ( dataByte >> ( 7 - ( bitIdx & 7 ) ) ) & 1
    crc = ( ( ( crc << 1 ) + bitVal ) & 0xFFFF ) ^ ( crcMsb * 0x1021 )
}
crcVal = crc

```

10

20

【0032】

`daih__atlas__b2p__checksum`は、`vuh__atlas__id`と関連付けられる復号アトラスのチェックサムである。`daih__atlas__b2p__checksum`の値は、以下のように得られる`checksumVal`の値に等しい。

```

checksum = 0
for( t = 0; t < daih_num_tiles_minus1 + 1; t++ ) {
    BlockToPatchMapWidth = Ceil( TileGroupWidth[ t ] / PatchPackingBlockSize )
    BlockToPatchMapHeight = Ceil( TileGroupHeight[ t ] / PatchPackingBlockSize )
    for( y = 0; y < BlockToPatchMapHeight; y++ ) {
        for( x = 0; x < BlockToPatchMapWidth; x++ ) {
            xorMask = ( x & 0xFF ) ^ ( y & 0xFF ) ^ ( x >> 8 ) ^ ( y >> 8 )
            checksum = ( checksum + ( BlockToPatchMap[ y ][ x ] & 0xFF ) ^ xorMask ) & 0xFFFFFFFF
            checksum = ( checksum + ( BlockToPatchMap[ y ][ x ] >> 8 ) ^ xorMask ) & 0xFFFFFFFF
        }
    }
}

```

30

40

【0033】

`daih_num_tiles_minus1 + 1`は、ハッシュ値がシグナリングされるタイルの数を特定する。`daih_num_tiles_minus1`の値は、`0 ~ afti_num_tiles_in_atlas_frame_minus1`の範囲内

50

である。

【0034】

`daih__tile__id__len__minus1 + 1`は、シンタックス要素`daih__tile__id[t]`を表すために使用されるビットの数を特定する。`daih__tile__id__len__minus1`の値は、 $0 \sim \text{Ceil}(\text{Log}2(\text{afti_num_tiles_in_atlas_frame_minus1} + 1))$ の範囲内である。 $1 << (\text{daih__tile__id__len__minus1} + 1)$ の値は、`daih__num__tile__minus1 + 1`よりも大きいか又はそれに等しい。

【0035】

`daih__tile__id[t]`は、 t 番目のタイルのタイルIDを特定する。`daih__tile__id[]`シンタックス要素の長さは、`daih__tile__id__len__minus1 + 1`ビットである。`daih__tile__id[t]`が存在しない場合には、その値が t に等しいと推測される。

10

【0036】

`daih__atlas__tiles__md5[t][i]`は、 t 番目のタイルの16バイトのMD5ハッシュである。`daih__atlas__tiles__md5[t][i]`の値は、IETF RFC 1321において定められるMD5関数を使用して以下のように得られる`digestVal`の値に等しい。

```
MD5Init(context)
```

20

```
MD5Update(context, tileData[t], dataLen[t])
```

```
MD5Final(digestVal, context)
```

【0037】

`daih__atlas__tiles__crc[t]`は、`vuh__atlas__id`と関連付けられる、タイルID t を含む復号アトラスタイルの周期冗長検査(CRC)である。`daih__atlas__tiles__crc`の値は、Rec. ITU-T H. 271において定められるCRC仕様を使用して以下のように得られる`crcVal`の値に等しい。

30

```
crc = 0xFFFF
```

```
tileData[t][dataLen[t]] = 0
```

```
tileData[t][dataLen[t] + 1] = 0
```

```
for(bitIdx = 0; bitIdx < (dataLen[t] + 2) * 8; bitIdx++) {
```

```
    dataByte = tileData[bitIdx >> 3]
```

```
    crcMsb = (crc >> 15) & 1
```

40

```
    bitVal = (dataByte >> (7 - (bitIdx & 7))) & 1
```

```
    crc = (((crc << 1) + bitVal) & 0xFFFF) ^ (crcMsb * 0x1021)
```

```
}
```

```
crcVal = crc
```

【0038】

`daih__atlas__tiles__checksum[t]`は、`vuh__atlas`

50

`__id`と関連付けられる、タイルID `t`を含む復号アタスタイルのチェックサムである。`daih__atlas__tile__checksum`の値は、以下のように得られる`checksumVal`の値に等しい。

```
checksum = 0
for(i = 0; i < dataLen[t]; i++) {
    xorMask = (i & 0xFF) ^ (i >> 8)
    checksum = (checksum + (tileData[t][i] & 0xFF) ^ xorMask) &
0xFFFFFFFF
    checksum = (checksum + (tileData[t][i] >> 8) ^ xorMask) &
0xFFFFFFFF
}
```

10

【0039】

`daih__atlas__tiles__b2p__md5[t][i]`は、`t`番目のタイル`BlockToPatchMap[ ][ ]`の16バイトのMD5ハッシュである。`daih__blk2patch__md5[t][i]`の値は、IETF RFC 1321において定められるMD5関数を使用して以下のように得られる`digestVal[t]`の値に等しい。

20

```
MD5Init(context)
MD5Update(context, tileB2pData[t], dataB2pLen[t])
MD5Final(digestVal, context)
```

30

【0040】

`daih__atlas__tiles__b2p__crc[t]`は、`vuh__atlas__id`と関連付けられる、タイルID `t`を含む復号アタスタイル`BlockToPatchMap[ ][ ]`の周期冗長検査(CRC)である。`daih__atlas__tiles__b2p__crc`の値は、Rec. ITU-T H. 271において定められるCRC仕様を使用して以下のように得られる`crcVal`の値に等しい。

40

50

```

crc = 0xFFFF
tileB2pData[t][dataB2pLen[t]] = 0
tileB2pData[t][dataB2pLen[t] + 1] = 0
for( bitIdx = 0; bitIdx < ( dataB2pLen[t] + 2 ) * 8; bitIdx++) {
    dataByte = tileB2pData[ bitIdx >> 3 ]
    crcMsb = ( crc >> 15 ) & 1
    bitVal = ( dataByte >> ( 7 - ( bitIdx & 7 ) ) ) & 1
    crc = ( ( ( crc << 1 ) + bitVal ) & 0xFFFF ) ^ ( crcMsb * 0x1021 )
}
crcVal = crc

```

10

【 0 0 4 1 】

`daih__atlas__tiles__b2p__checksum[t]` は、`vuh__atlas__id` と関連付けられる、タイル ID `t` を含む復号アトラスタイル `BlockToPatchMap[ ][ ]` のチェックサムである。`daih__atlas__tile__checksum` の値は、以下のように得られる `checksumVal` の値に等しい。

20

```

checksum = 0
BlockToPatchMapWidth = Ceil(TileGroupWidth[t]/PatchPackingBlockSize)
BlockToPatchMapHeight = Ceil(TileGroupHeight[t]/PatchPackingBlockSize)
for( y = 0; y < BlockToPatchMapHeight; y++) {
    for( x = 0; x < BlockToPatchMapWidth; x++) {
        xorMask = ( x & 0xFF ) ^ ( y & 0xFF ) ^ ( x >> 8 ) ^ ( y >> 8 )
        checksum = ( checksum + ( BlockToPatchMap[y][x] & 0xFF ) ^ xorMask ) & 0xFFFFFFFF
        checksum = ( checksum + ( BlockToPatchMap[y][x] >> 8 ) ^ xorMask ) & 0xFFFFFFFF
    }
}

```

30

40

【 0 0 4 2 】

本明細書では、復号アトラス情報ハッシュ `SEI` メッセージについて、そのシンタックス及びセマンティクスと共に説明する。適合点 A におけるブロックからボリュームへの情報の適切な復号、及びボリュメトリックデータの正しい再構成を確実にするために、“`BlockToPatch`” ハッシュ値も含まれる。チェックサム計算の `md5` 方法に加えて、潜在的な拡張性のために、他の 2 つのハッシュタイプ、すなわち、`CRC` 及びチェックサムが列挙される。`V3C/V-PCC` 仕様に、アネックス B (`Annex B`) のようなセクションを含み、ハッシュチェックサム値の公称導出のための例を提供することができる。ハッシュメッセージのチェックサム計算のための「フレームヘッダ」として、アネックス B において再構成セクション H. 9 のために使用される高レベルのシンタックス要素

50

に関連する複数の変数を含むことができる。

【0043】

図1に、いくつかの実施形態による、コンテンツを符号化及び復号する方法のフローチャートを示す。ステップ100において、ポイントクラウドコンテンツを符号化する。コンテンツを符号化するステップは、3Dから2Dへの投影に基づいてポイントクラウドを圧縮するV-PCC符号化実装を使用するステップなどの任意の方法で実装することができる。この方法は、3Dポイントクラウドデータを複数の2Dパッチにマッピングした後、更にこれらのパッチをアトラス画像内に配置し、その後にビデオエンコーダで符号化する。アトラス画像は、ポイントのジオメトリ、それぞれのテクスチャ、及びどの位置をポイントクラウド再構成のために検討すべきであることを示す占有マップに対応する。MP

10

【0044】

ステップ102において、ハッシュを生成して送信する。本明細書で説明するように、ハッシュは、コンテンツのシンタックス要素及び/又は変数値を使用して生成される。いくつかの実施形態では、ハッシュは、アトラスに対して又はアトラスのblock to patchに対して送信される。いくつかの実施形態では、ハッシュは、タイルに対して又はタイルのblock to patchに対して送信される。ハッシュは、SEIメッセージで送信される。

20

【0045】

ステップ104において、符号化されたコンテンツ（ポイントクラウド）を復号する。MP

【0046】

ステップ106において、ハッシュ情報を使用して、復号されたコンテンツを検証する。復号されたコンテンツは、任意のハッシュ比較技術を介して、ハッシュ情報を使用して検証される。

【0047】

いくつかの実施形態では、より少ない又は追加のステップを実装することができる。いくつかの実施形態では、ステップの順序が変更される。

30

【0048】

図2に、いくつかの実施形態による、ハッシュを生成するフローチャートを示す。予め、ハッシュは、輝度、色及び/又は他の画素値に基づいている。ステップ200において、コンテンツ（例えばアトラス又はタイル）のシンタックス要素又は他の情報の値を組み合わせることによって、バイト列を生成する。例えば、アトラスの場合、フレーム幅を2バイト（又は何らかの他の値）にする。次に、フレーム高さは次の2バイトである。この列に（アトラスの）他のシンタックス要素及び変数値を追加して、大きい列（例えば256バイト）を生成する。ステップ202において、バイト列にハッシュアルゴリズムを適用して、ハッシュを生成する。ハッシュアルゴリズムは、定義された変数に基づいて選択することができる。例えば、ルックアップテーブル又はデータベースが、複数のハッシュアルゴリズム又は同様のアルゴリズム（例えば周期冗長検査）を記憶することができ、定義された変数に基づいて、選択されたハッシュアルゴリズムを適用する。本明細書で説明するように、0の変数値はハッシュアルゴリズムMD5に対応し、1の変数値はCRCに対応し、2の変数値はチェックサムに対応し、一方、追加のハッシュアルゴリズムのために、追加の変数値を留保することができる。いくつかの実施形態では、より少ない又は追加のステップが実装される。いくつかの実施形態では、ステップの順序が変更される。

40

【0049】

図3に、いくつかの実施形態による、ハッシュを実装するように構成されるシステムの

50

図を示す。エンコーダ300が、符号化プロセスを実装するように構成される。本明細書で説明するように、1又は2以上のMPEG実装を使用して、V-PCC符号化などの任意の符号化を実装することができる。更に、エンコーダは、符号化されたコンテンツと通信されるべきハッシュ情報を生成する。符号化された情報及びハッシュ情報は、デコーダ304に直接又はネットワーク302を通じて通信することができる。ネットワークは、ローカルエリアネットワーク(LAN)、インターネット、無線ネットワーク、有線ネットワーク、セルラネットワーク、及び／又は他の任意のネットワーク又はネットワークの組み合わせなどの任意のタイプのネットワークとすることができる。デコーダ304は、符号化されたコンテンツを復号し、ハッシュ情報を使用して、復号されたコンテンツを検証する。

【0050】

図4に、いくつかの実施形態による、復号タイルハッシュSEIメッセージを実装するように構成される例示的なコンピュータ装置のブロック図を示す。コンピュータ装置400を使用して、3Dコンテンツを含む画像及びビデオなどの情報を取得、記憶、計算、処理、通信及び／又は表示することができる。コンピュータ装置400は、符号化／復号の態様のいずれかを実装することができる。一般に、コンピュータ装置400を実装するのに適したハードウェア構造は、ネットワークインターフェイス402、メモリ404、プロセッサ406、I/Oデバイス408、バス410、及び記憶装置412を含む。プロセッサの選択は、十分な速度を有する好適なプロセッサが選ばれる限り重要ではない。メモリ404は、当該技術分野で公知の任意の従来のコンピュータメモリとすることができる。記憶装置412は、ハードドライブ、CDROM、CDRW、DVD、DVDRW、高精細ディスク／ドライブ、超高精細ドライブ、フラッシュメモリカード又は他の任意の記憶装置を含むことができる。コンピュータ装置400は、1又は2以上のネットワークインターフェイス402を含むことができる。ネットワークインターフェイスの一例は、イーサネット又は他のタイプのLANに接続されるネットワークカードを含む。I/Oデバイス408は、以下のもの、すなわち、キーボード、マウス、モニタ、スクリーン、プリンタ、モデム、タッチスクリーン、ボタンインターフェイス及び他のデバイスのうちの1又は2以上を含むことができる。復号タイルハッシュSEIメッセージを実装するのに使用される復号タイルハッシュSEIメッセージアプリケーション430は、記憶装置412及びメモリ404に記憶されて、アプリケーションが通常処理されるように処理される可能性が高い。コンピュータ装置400は、図4に示すより多い又は少ない構成要素を含むことができる。いくつかの実施形態では、復号タイルハッシュSEIメッセージハードウェア420が含まれる。図4のコンピュータ装置400は、復号タイルハッシュSEIメッセージのためのアプリケーション430及びハードウェア420を含むが、復号タイルハッシュSEIメッセージは、コンピュータ装置に、ハードウェア、ファームウェア、ソフトウェア又はそれらの任意の組み合わせとして実装することができる。例えば、いくつかの実施形態では、復号タイルハッシュSEIメッセージアプリケーション430は、メモリにプログラムされて、プロセッサを使用して実行される。別の例では、いくつかの実施形態では、復号タイルハッシュSEIメッセージハードウェア420は、復号タイルハッシュSEIメッセージを実装するように専用に設計されるゲートを含む、プログラムされたハードウェアロジックである。

【0051】

いくつかの実施形態では、復号タイルハッシュSEIメッセージアプリケーション430は、複数のアプリケーション及び／又はモジュールを含む。いくつかの実施形態では、モジュールは、1又は2以上のサブモジュールも含む。いくつかの実施形態では、より少ない又は追加のモジュールを含むことができる。

【0052】

好適なコンピュータ装置の例は、パーソナルコンピュータ、ラップトップコンピュータ、コンピュータワークステーション、サーバ、メインフレームコンピュータ、ハンドヘルドコンピュータ、携帯情報端末、セルラ電話／携帯電話、スマート家電、ゲーム機、デジ

10

20

30

40

50

タルカメラ、デジタルカムコーダ、カメラ付き携帯電話、スマートフォン、携帯音楽プレーヤー、タブレットコンピュータ、モバイルデバイス、ビデオプレーヤー、ビデオディスクライター／プレーヤー（例えば、DVDライター／プレーヤー、高精細ディスクライター／プレーヤー、超高精細ディスクライター／プレーヤー）、テレビジョン、家庭用娯楽システム、拡張現実デバイス、仮想現実デバイス、スマートジュエリー（例えば、スマートウォッチ）、車両（例えば、自動運転車両）又は他の任意の好適なコンピュータ装置を含む。

【0053】

復号タイルハッシュSEIメッセージを利用するために、装置が3Dコンテンツ（例えば、ポイントクラウドコンテンツ）を取得又は受信し、ハッシュ情報を含むSEIメッセージを含むコンテンツを処理及び／又は送信して、3Dコンテンツが適切に復号されることを確実にする。復号タイルハッシュSEIメッセージは、ユーザの援助によって又はユーザが関与することなく自動的に実装することができる。

10

【0054】

動作時、復号タイルハッシュSEIメッセージは、従来の実装と比較して、より効率的でより正確な3Dコンテンツ符号化を可能にする。

【0055】

V3C／V-PCCのための復号タイルハッシュSEIメッセージのいくつかの実施形態

1. ポイントクラウドコンテンツを符号化して、符号化されたポイントクラウドコンテンツを生成するステップと、

20

ハッシュ情報を生成して送信するステップと、

前記符号化されたポイントクラウドコンテンツを復号するステップと、

前記ハッシュ情報を使用して、前記復号されたコンテンツを検証するステップと、を含む方法。

【0056】

2. 前記ポイントクラウドコンテンツを符号化するステップは、3Dから2Dへの投影に基づいてポイントクラウドを圧縮するV-PCC符号化実装を使用するステップを含み、前記ポイントクラウドコンテンツを符号化するステップは、3Dポイントクラウドデータを複数の2Dパッチにマッピングし、前記パッチをアトラス画像内に配置し、その後ビデオエンコーダで符号化するステップを含み、アトラス画像は、ポイントのジオメトリ、それぞれのテクスチャ、及びどの位置をポイントクラウド再構成のために検討すべきであることを示す占有マップに対応する、第1項に記載の方法。

30

【0057】

3. 前記ハッシュ情報は、前記ポイントクラウドコンテンツのシンタックス要素及び／又は変数値を使用して生成される、第1項に記載の方法。

【0058】

4. 前記ハッシュ情報は、アトラスに対して送信される、第1項に記載の方法。

【0059】

5. 前記ハッシュ情報は、アトラスのblock to patchに対して送信される、第1項に記載の方法。

40

【0060】

6. 前記ハッシュ情報は、タイルに対して送信される、第1項に記載の方法。

【0061】

7. 前記ハッシュ情報は、タイルのblock to patchに対して送信される、第1項に記載の方法。

【0062】

8. 前記ハッシュ情報は、補足強化情報（SEI）メッセージで送信される、第1項に記載の方法。

【0063】

9. 前記SEIメッセージは、タイルベースの“BlockToPatchMap []

50

[] ”ハッシュ値情報を含む、第 8 項に記載の方法。

【0064】

10. 前記ハッシュ情報は、MD5、CRC 及びチェックサムからなるグループから選択される実装を使用して生成される、第 1 項に記載の方法。

【0065】

11. 符号化されたポイントクラウドコンテンツを受け取り、
ハッシュ情報を受け取り、
前記符号化されたポイントクラウドコンテンツを復号し、
前記ハッシュ情報を使用して、前記復号されたコンテンツを検証する、
ためのアプリケーションを記憶するための非一時的メモリと、
前記メモリに結合され、前記アプリケーションを処理するように構成されるプロセッサと、
を備える装置。

10

【0066】

12. 前記ハッシュ情報は、前記ポイントクラウドコンテンツのシンタックス要素及び/又は変数値を使用して生成される、第 11 項に記載の装置。

【0067】

13. 前記ハッシュ情報は、アトラスに対して送信される、第 11 項に記載の装置。

【0068】

14. 前記ハッシュ情報は、アトラスの `block to patch` に対して送信される、第 11 項に記載の装置。

20

【0069】

15. 前記ハッシュ情報は、タイルに対して送信される、第 11 項に記載の装置。

【0070】

16. 前記ハッシュ情報は、タイルの `block to patch` に対して送信される、第 11 項に記載の装置。

【0071】

17. 前記ハッシュ情報は、補足強化情報 (SEI) メッセージで送信される、第 11 項に記載の装置。

【0072】

18. 前記 SEI メッセージは、タイルベースの “BlockToPatchMap [] [] ” ハッシュ値情報を含む、第 17 項に記載の装置。

30

【0073】

19. 前記ハッシュ情報は、MD5、CRC 及びチェックサムからなるグループから選択される実装を使用して生成される、第 11 項に記載の装置。

【0074】

20. 3 次元コンテンツを取得するための 1 又は 2 以上のカメラと、
前記 3 次元コンテンツを符号化して、符号化されたポイントクラウドコンテンツを生成し、

ハッシュ情報を生成して送信する、
ためのエンコーダと、

40

前記符号化されたポイントクラウドコンテンツを復号し、
前記ハッシュ情報を使用して、前記復号されたコンテンツを検証する、
ためのデコーダと、
を備えるシステム。

【0075】

21. 前記 3 次元コンテンツを符号化することは、3D から 2D への投影に基づいて 3 次元コンテンツを圧縮する V-PCC 符号化実装を使用することを含み、前記 3 次元コンテンツを符号化することは、3D ポイントクラウドデータを複数の 2D パッチにマッピングし、前記パッチをアトラス画像内に配置し、その後にビデオエンコーダで符号化するこ

50

とを含み、アトラス画像は、ポイントのジオメトリ、それぞれのテクスチャ、及びどの位置をポイントクラウド再構成のために検討すべきであるかを示す占有マップに対応する、第20項に記載のシステム。

【0076】

22. 前記ハッシュ情報は、前記3次元コンテンツのシンタックス要素及び/又は変数値を使用して生成される、第20項に記載のシステム。

【0077】

23. 前記ハッシュ情報は、アトラスに対して送信される、第20項に記載のシステム。

【0078】

24. 前記ハッシュ情報は、アトラスの `block to patch` に対して送信される、第20項に記載のシステム。

【0079】

25. 前記ハッシュ情報は、タイルに対して送信される、第20項に記載のシステム。

【0080】

26. 前記ハッシュ情報は、タイルの `block to patch` に対して送信される、第20項に記載のシステム。

【0081】

27. 前記ハッシュ情報は、補足強化情報 (SEI) メッセージで送信される、第20項に記載のシステム。

【0082】

28. 前記SEIメッセージは、タイルベースの“`BlockToPatchMap [ ] [ ]`”ハッシュ値情報を含む、第27項に記載のシステム。

【0083】

29. 前記ハッシュ情報は、MD5、CRC及びチェックサムからなるグループから選択される実装を使用して生成される、第20項に記載のシステム。

【0084】

本発明の構成及び動作の原理の理解を容易にするために、詳細内容を組み込んだ特定の実施形態に関して本発明を説明してきた。このような本明細書における特定の実施形態及びその詳細内容への言及は、本明細書に添付される特許請求の範囲を限定することを意図するものではない。特許請求の範囲によって規定される本発明の精神及び範囲から逸脱することなく、例示のために選択された実施形態に、他の様々な修正を行うことができることは、当業者に容易に理解されるであろう。

【符号の説明】

【0085】

100 ポイントクラウドを符号化

102 ハッシュを生成して送信

104 符号化されたポイントクラウドを復号

106 ハッシュを使用して、復号されたポイントクラウドを検証

200 バイト列を生成

202 バイト列にハッシュアルゴリズムを適用

300 エンコーダ

302 ネットワーク

304 デコーダ

400 コンピュータ装置

402 ネットワークインターフェイス

404 メモリ

406 プロセッサ

408 I/Oデバイス

410 バス

412 記憶装置

4 2 0 復号タイルハッシュ S E I メッセージハードウェア
4 3 0 復号タイルハッシュ S E I メッセージアプリケーション

【図面】

【図 1】

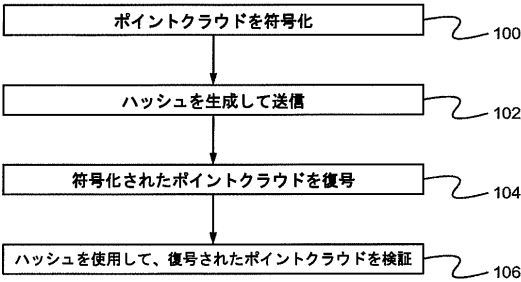


Fig. 1

【図 2】

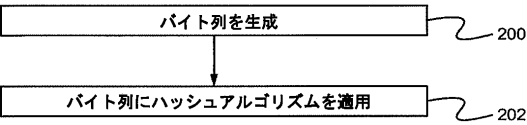


Fig. 2

【図 3】

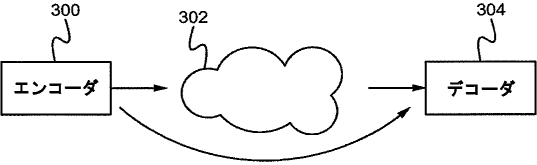


Fig. 3

【図 4】

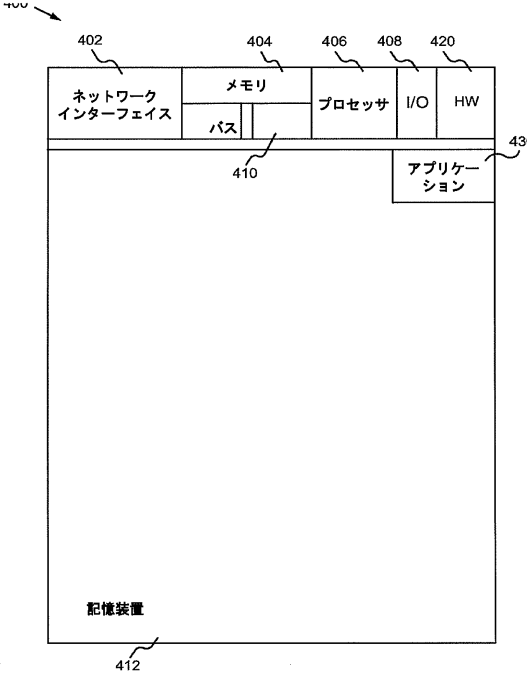


Fig. 4

10

20

30

40

50

フロントページの続き

- (33)優先権主張国・地域又は機関
米国(US)
- (31)優先権主張番号 63/046,749
- (32)優先日 令和2年7月1日(2020.7.1)
- (33)優先権主張国・地域又は機関
米国(US)
- (31)優先権主張番号 17/161,319
- (32)優先日 令和3年1月28日(2021.1.28)
- (33)優先権主張国・地域又は機関
米国(US)
フラッグスタッフ サウス マジェスティック ロード 2200
- (72)発明者 タバタバイ アリ
アメリカ合衆国 カリフォルニア州 95014 クパチーノ イースト エステーツ ドライヴ 10265
- 審査官 間宮 嘉誉
- (56)参考文献 国際公開第2021/201928 (WO, A1)
米国特許出願公開第2020/0195946 (US, A1)
米国特許出願公開第2015/0271529 (US, A1)
米国特許出願公開第2016/0277761 (US, A1)
特表2023-519512 (JP, A)
- (58)調査した分野 (Int.Cl., DB名)
G06T 9/00-9/40
H04N 7/12
H04N 19/00-19/98