

(19) 日本国特許庁 (JP)

(12) 特 許 公 報 (B2)

(11) 特許番号

特許第5591584号  
(P5591584)

(45) 発行日 平成26年9月17日 (2014. 9. 17)

(24) 登録日 平成26年8月8日 (2014. 8. 8)

(51) Int. Cl. F I  
**HO 4 N 19/467 (2014. 01)** HO 4 N 19/467  
**HO 4 N 19/70 (2014. 01)** HO 4 N 19/70

請求項の数 5 外国語出願 (全 23 頁)

(21) 出願番号	特願2010-112487 (P2010-112487)	(73) 特許権者	501263810
(22) 出願日	平成22年5月14日 (2010. 5. 14)		トムソン ライセンシング
(65) 公開番号	特開2010-279031 (P2010-279031A)		Thomson Licensing
(43) 公開日	平成22年12月9日 (2010. 12. 9)		フランス国, 92130 イッシー レ
審査請求日	平成25年2月27日 (2013. 2. 27)		ムーリノー, ル ジャンヌ ダルク,
(31) 優先権主張番号	09305493.0		1-5
(32) 優先日	平成21年5月29日 (2009. 5. 29)		1-5, rue Jeanne d' A
(33) 優先権主張国	欧州特許庁 (EP)		rc, 92130 ISSY LES
			MOULINEAUX, France
		(74) 代理人	100070150
			弁理士 伊東 忠彦
		(74) 代理人	100091214
			弁理士 大貫 進介
		(74) 代理人	100107766
			弁理士 伊東 忠重

最終頁に続く

(54) 【発明の名称】 ビットストリームに透かし支援データを挿入する方法、及び透かし支援データを含むビットストリーム

(57) 【特許請求の範囲】

【請求項 1】

ディジタル・コンテンツを表す符号化データのビットストリームに透かし支援データを挿入する方法であって、

\_\_置換する対象の符号化データを識別する識別データを前記符号化データのビットストリームに挿入する工程と、

置換データの数と、置換が行われるやり方と、前記符号化データの置換が誤差検査に従って行われるか否かとを規定するフォーマット・データを前記符号化データのビットストリームに挿入する工程と、

置換する対象の符号化データ毎に、前記フォーマット・データに応じて、置換する対象の前記符号化データが送信中に何らかの修正を受けたか否かを検査するために、検査データと少なくとも1つの置換データとを前記符号化データのビットストリームに挿入する工程と

を含み、

前記識別データ、前記フォーマット・データ、前記少なくとも1つの置換データ及び前記検査データは、補助拡張情報メッセージの形式で前記符号化データのビットストリームに挿入される方法。

【請求項 2】

請求項 1 記載の方法であって、前記検査データは、置換する対象の前記符号化データから計算される巡回冗長度検査である方法。

10

20

## 【請求項 3】

請求項 1 記載の方法であって、前記検査データは、置換する対象の前記符号化データである方法。

## 【請求項 4】

請求項 1 記載の方法であって、前記識別データが、置換する対象の前記符号化データの先頭を識別する絶対アドレスと、前記置換データの長さとを含む方法。

## 【請求項 5】

請求項 1 記載の方法であって、前記符号化データのビットストリームが伝送パケットにカプセル化され、前記識別データが、置換する対象の前記符号化データの先頭を識別する前記伝送パケットの先頭に対するオフセット値と、置換する対象の前記符号化データを含む伝送パケットを識別するデータと、前記置換データの長さとを含むことを規定する方法。

10

## 【発明の詳細な説明】

## 【技術分野】

## 【0001】

本発明は、符号化の技術分野に関する。より具体的には、本発明は、符号化データ・ビットストリームに透かし支援データを挿入する方法に関し、挿入された透かし支援データを使用して、符号化データ・ストリームに透かしを入れる方法に関し、透かしを入れる方法を実現する装置に関する。本発明は更に、挿入方法によって挿入された透かし支援データを含む符号化データ・ビットストリームに関する。

20

## 【背景技術】

## 【0002】

デジタル・コンテンツ（例えば、ビデオ、オーディオや 3D データ等）を保護するために、認可なしでコンテンツの伝送を認可されている個人又は組織体を識別するように配信されたストリームそれぞれに、デジタル符号のフォーマットで生じる一意の透かしを挿入することが知られている。例として、映画の販売促進中に、別々の透かしを利用して透かしを入れた DVD が、選択された個人に供給される。漏洩の場合、漏洩の元を識別することが、透かしを再構成することによって可能である。他の適用例も考えられる。すなわち、透かしを挿入し、それにより、作品又は受益者を識別するか、又は、さもなければ、透かしにより、補助データ（メタデータ）を伝送することが可能になる。

30

## 【0003】

この目的で、ビデオ透かし入れは、画像のビデオ / 系列を表す符号化データのビットストリームを個人化するために消費者向装置によって使用されることが知られている。「個人化」は、符号化データや、他の種類の情報（著作権など）のビットストリーム中の、一例として、「ユーザ ID」及び、場合によっては、タイム・スタンプを表すデジタル符号を挿入することによって達成される。本願では、「透かし」、「デジタル符号」及び「ユーザ ID」は、符号化データ・ビットストリームに挿入されるデジタル符号を表すために同義で使用される。前述の個人化を行うために、消費者向装置において必要な計算能力を低減させるために、透かし置換マップ（WSM と呼ばれる）などの透かし支援データの「支援」による「置換による透かし」手法を使用することが知られている。前述の支援された置換手法は、3 工程において透かし入れ処理を実現する。

40

第 1 の工程は、画像系列、又は、より一般には、放送される対象のデジタル・コンテンツを使用して、透かし支援データが生成される放送業者のヘッドエンドにおいて行われる。次いで、第 2 の工程中に、透かし支援データは、符号化データのビットストリームに埋め込まれ、第 3 の工程が行われる消費者側に送信される。第 3 の工程中に、透かしを入れることにより、符号化データの受信されたビットストリームの個人化を支援するよう、透かし支援データが消費者向装置において使用される。第 1 の工程の目的は、透かし（すなわち、個人化）を確実にするために、置換が、符号化データのビットストリームを損なわないということを確実にし、透かし（すなわち、個人化）が、画像系列の再構成後、消費者によって可視 / 可聴でないということを確実にし、関連する特定の攻撃に対して充分

50

ロバストであるということを確実にする。したがって、第1の工程は、符号化データのビットストリームにおける、置換する対象のデータを識別する工程、及び置換データを求める工程を含む。第1の工程は更に、ロバスト性及び忠実度を保証する。透かし支援データは、前述の第1の工程から生じる。

【0004】

第2の工程では、次いで、透かし支援データを、消費者向装置に送信する対象の符号化データ・ビットストリームに埋め込む。従来技術では、MPEG2-TSの私用データ又はユーザ・データ(TSは「伝送ストリーム」を意味する)を使用して透かし支援データを埋め込むことが知られている。MPEG2-TS伝送層は、「Information Technology - Generic Coding of moving images and associated audio Systems ITU-T Rec. H.222.0 | ISO/IEC 13818-1 (2<sup>nd</sup> edition, 2000)」と題する文献において規定されている。

10

【0005】

第3の工程では、個人化透かし入れは、圧縮ドメインにおいて消費者向装置によって施される。実際に、透かしは、符号化データ・ビットストリームに直接「挿入される」。前述の挿入は、埋め込まれた透かし支援データに含まれる情報に基づいて、置換データと呼ばれる、特定の他のものにより、特定の符号化データを符号化データ・ビットストリームにおいて置換する(置き換える)ことによって行われる。圧縮ドメインにおいて透かしを挿入することにより、ビットストリーム自体、及び符号化データのビットストリームの復号化によって生じるベースバンド・コンテンツを何れも保護することが可能になる。実際に、透かしは、デコンプレッション中に、ベースバンド・コンテンツに伝播する。

20

【0006】

前述の解決策の第1の欠点は、MPEG2-TSの私用データ又はユーザ・データが容易に除去可能であり、これにより、攻撃に対する上記解決策のロバスト性が低くなるという点である。第2に、MPEG2-TSの私用データ及びユーザ・データは暗号化されず、よって、特定の更なる暗号化機構が必要になる。更に、私用データに割り当てられるMPEG2-TSパケットの一部分の容量は、容量が制限され、それにより、透かし支援データの容量が制限される。更に、ユーザ・データでは、画像系列との透かし支援データの同期化は単純でない。実際に、ユーザ・データはMPEG2-TSパケットにカプセル化され、これは、ビデオをカプセル化し、よって、種々のパケットの同期化を要求するものとは異なる。最後に、従来技術において規定されたWSMは、種々の適用例要件に対する柔軟性をもたらさない。

30

【発明の概要】

【発明が解決しようとする課題】

【0007】

本発明は、従来技術の欠点のうちの少なくとも1つを軽減することを目的としている。

【課題を解決するための手段】

【0008】

前述の目的のために、本発明は、符号化データ・ビットストリームに透かし支援データを挿入する方法に関する。上記方法は、置換する対象の符号化データを識別する識別データを符号化データ・ビットストリームに挿入し、少なくとも1つの置換データを、置換する対象の符号化データ毎に挿入する工程を含む。上記方法は、置換データによって置換される対象の符号化データを置換するやり方を規定するフォーマット・データを符号化データ・ビットストリームに挿入する工程を更に含む。フォーマット・データは、種々のアプリケーション要件に対する最善の一致を可能にするための柔軟性を提供する。

40

【0009】

特定の実施例によれば、上記方法は、検査データを、置換する対象の符号化データ毎に、符号化データ・ビットストリームに挿入する工程を更に含む。

【0010】

50

特定の特徴によれば、検査データは、置換する対象の符号化データから計算される巡回冗長度検査である。

【 0 0 1 1 】

変形によれば、検査データは、置換する対象の符号化データである。

【 0 0 1 2 】

特定の特徴によれば、識別データは、置換する対象の符号化データの先頭を識別する絶対アドレスである。

【 0 0 1 3 】

符号化データ・ビットストリームが伝送パケットにおいてカプセル化される場合に使用される変形によれば、識別データは、置換する対象の符号化データの先頭を識別する伝送パケットの先頭に対するオフセット値、及び置換する対象の符号化データを含む伝送パケットを識別するデータを含む。

10

【 0 0 1 4 】

特定の特徴によれば、識別データ、置換符号化データ、及びフォーマット・データが、補助拡張情報メッセージのフォーマットで符号化データ・ビットストリームに挿入される。

【 0 0 1 5 】

本発明は、更に、符号化データのビットストリームに関し、上記符号化データは、置換する対象の符号化データを識別する識別データと、置換する対象の符号化データ毎の少なくとも1つの置換データと、置換データによって置換される対象の符号化データを置換するやり方を規定するフォーマット・データとを含む。

20

【 0 0 1 6 】

更に、本発明は、

置換データによって置換する対象の符号化データを置換するやり方を規定するフォーマット・データ、置換する対象の符号化データ毎の少なくとも1つの置換データ、及び置換する対象の符号化データを識別する識別データを符号化データ・ビットストリームから判定する工程と、

識別データにより、置換する対象の符号化データを識別する工程と、

フォーマット・データ及び透かしデータに基づいて、1つの置換符号化データで置換される対象の符号化データそれぞれを置換することにより、符号化データ・ビットストリームに透かしを入れる工程とを含む、透かしにより、符号化データ・ビットストリームに透かしを入れる方法を提案している。

30

【 0 0 1 7 】

本発明は、更に、

デジタル・コンテンツを表す符号化データ・ビットストリームを受信するための入力部と、

装置に関連付けられた識別子に基づいて符号化データ・ビットストリームを、透かしを入れたビットストリームに、透かしを入れるための透かし入れモジュールと、

透かしを入れたビットストリームから、透かしを入れたデジタル・コンテンツを再構成するための復号化モジュールと、

40

透かしを入れたビットストリームを送信するための第1の出力部と、

透かしを入れたデジタル・コンテンツを送信するための第2の出力部とを備える装置に関する。

【 0 0 1 8 】

装置の透かし入れモジュールは、

置換データによって置換する対象の符号化データを置換するやり方を規定するフォーマット・データ、置換する対象の符号化データ毎の少なくとも1つの置換データ、及び置換する対象の符号化データを識別する識別データを符号化データ・ビットストリームから判定する装置と、

50

識別データにより、置換する対象の符号化データを識別する装置と、

フォーマット・データ及び透かしデータに基づいて、1つの置換符号化データで置換される対象の符号化データを置換することにより、符号化データ・ビットストリームに透かしを入れる透かし入れ装置とを備える。

【図面の簡単な説明】

【0019】

【図1】本発明による、挿入する方法を示すブロック図である。

【図2】本発明によって規定される種々の置換データ及び符号化データのビットストリームを表す図である。

【図3】本発明によって規定される種々の置換データ及び符号化データのビットストリームを表す図である。

10

【図4】本発明によって規定される種々の置換データ及び符号化データのビットストリームを表す

【図5】本発明による、透かしを入れる方法を示すブロック図である。

【図6】本発明による、符号化データ・ビットストリーム、及び透かしを入れる方法によって透かしを入れたその関連したビットストリームを示す図である。

【図7】本発明による装置を表す図である。

【発明を実施するための形態】

【0020】

本発明の他の構成及び利点は、その実施例の一部の以下の説明によって明らかになるであろう。前述の説明は、図面に関して行う。

20

【実施例】

【0021】

本発明は、符号化データ・ビットストリームへの(WSMなどの)透かし支援データの挿入の方法に関する。話を単純にするために、符号化データ・ビットストリームは、以下では、画像系列を表すものとする。しかし、符号化データ・ビットストリームは、オーディオ・データや3Dデータなどの何れかの種類のデジタル・コンテンツを表し得る。一般に、符号化データ・ビットストリームとしての系列画像の符号化は、先行符号化画像の画像データからの画像データの予測(時間予測)、又は先行符号化された同じ画像の画像データからの画像データの予測(空間予測)を含む。元のデータから予測データを減算することによって計算される残差データは一般に、(例えば、離散コサイン変換、又はその近似を使用して)変換され、量子化される。量子化データは最終的に、例えば、VLC(「可変長符号」の頭字語)テーブルを使用して符号化データ・ビットストリームにエントロピ符号化される。例として、系列は、「Information technology - Coding of audio-visual objects - Part 10: Advanced Video Coding」と題する西暦2005年2月15日付に公表されたISO/IEC 13818-2に記載されたMPEG-2標準によって符号化される。本発明は、標準によっても、デジタル・コンテンツの種類によっても制限されない。

30

【0022】

本発明によれば、透かし支援データはいくつかのエンティティを有し得る。各エントリは、

40

置換する対象の符号化データを符号化データ・ビットストリームにおいて識別するための少なくとも1つの識別データと、

少なくとも1つの置換データとを含む。

【0023】

例として、識別データは、符号化データ・ビットストリームの先頭から、置換する対象の符号化データの先頭(例えば、第1のビット又はバイト)までのオフセット値として規定される。

50

## 【 0 0 2 4 】

変形によれば、識別データは、画像の先頭から、置換する対象の符号化データの先頭までのオフセット値として規定される。

## 【 0 0 2 5 】

変形によれば、透かし支援データが、MPEG-2 TS パケットなどの伝送パケットにカプセル化された場合、オフセット値は、

置換する対象の符号化データの第1のデータ（例えば、第1のビット又はバイト）を含むパケットに対するレファレンス、及び

前述のパケットの先頭から、置換する対象の符号化データの第1のデータまでのオフセット値

10

という2つの部分を含む。

## 【 0 0 2 6 】

例として、置換する対象の最大「m」個の符号化データが、符号化データ・ビットストリーム内の「n」個の画像毎に識別され、「m」及び「n」は16や1などの整数である。変形によれば、置換する対象の前述の符号化データは、西暦2008年10月2日に公開された国際公開2008/118145号パンフレットに開示された方法によって識別される。例として、置換データは、西暦2008年10月2日に公開された国際公開2008/118145号パンフレットに開示された方法によって求められる。

## 【 0 0 2 7 】

符号化データのビットストリームFに、透かし支援データを挿入する方法の第1の例示的な実施例は図1を参照して説明する。符号化データ・ビットストリームは以下では画像系列を表すものであるが、オーディオ・データや3Dデータなどの何れかの種類のデジタル・コンテンツを表し得る。

20

## 【 0 0 2 8 】

工程12では、置換する対象の符号化データを識別する少なくとも1つの識別データ、及び少なくとも1つの置換データが符号化データ・ビットストリームに挿入される。上記識別データ及び置換データは、符号化データ・ビットストリームから計算するか、又はルックアップ・テーブルによって提供することが可能である。

## 【 0 0 2 9 】

工程14では、置換される対象の符号化データを置換データによって置換するやり方を規定するフォーマット・データは、符号化データFのビットストリームに挿入される。

30

## 【 0 0 3 0 】

フォーマット・データは、ユーザ・インタフェースなどの外部手段により、パラメータとして提供することが可能である。前述のフォーマット・データは、アプリケーションに応じて固定してもよく、又は動的に変更してもよい。フォーマット・データの選択は、

符号化データFのビットストリームを生成するために使用されている圧縮手法及び、それが置換を可能にする能力、

圧縮手法の符号化データ構文のビットストリーム、

画像毎にアプリケーションが行うことが必要な置換の数、

一部の特定のフォーマットによって最小にすることが可能な余剰ペイロードを、透かし  
思念データに割り当てることが可能なビットレート、

40

デコンプレッション誤差を避ける必要性、

透かしを標的とした攻撃に対するロバスト性、

埋め込む対象のデジタル符号の実際のサイズ、及び

経時的な反復率等

のいくつかの局面間のトレードオフである。

## 【 0 0 3 1 】

フォーマット・データは、種々のアプリケーション要件に対する最善の一致を可能にするための柔軟性を提供する。よって、多くの変形を導き出すことが可能である。

## 【 0 0 3 2 】

50

例として、フォーマット・データは、置換データ数を規定し、誤差検査に従って置換を行うかを規定する。符号化データ・ビットストリームにおいて符号化されたフォーマット・データに応じて、特定の更なるデータ（検査データなど）を符号化することができる。

#### 【0033】

例として、第1の置換フォーマット（フォーマット1）によれば、2つの置換データが工程12で挿入される。符号化データ・ビットストリームに「0」を埋め込むために置換する対象の符号化データを置換するために、消費者向装置において、置換データD1のうちの1つが使用され、符号化データ・ビットストリームに「1」を埋め込むために置換する対象の符号化データを置換するために、置換データD2のうちの第2のものが使用される。このケースは、図2の左部分に示す。

10

#### 【0034】

第2の置換フォーマット（フォーマット2）では、工程12で、単一の置換データが挿入される。符号化データ・ビットストリームに「1」を埋め込むために、置換する対象の符号化データがそれ自体によって置換される（すなわち、置換は達成されない）一方で、符号化データ・ビットストリームに「0」を埋め込むために置換する対象の符号化データを置換するために、単一の置換データD1が消費者向装置において使用される。変形によれば、符号化データ・ビットストリームに「0」を埋め込むために、置換する対象の符号化データがそれ自体によって置換される（すなわち、置換は達成されない）一方で、符号化データ・ビットストリームに「1」を埋め込むために置換する対象の符号化データを置換するために、単一の置換データが使用される。このケースは、図2の右部分に示す。

20

#### 【0035】

置換の第1のフォーマット及び第2のフォーマットの変形によれば、検査データが工程12で挿入される。前述のケースは図3に示す。検査データは例えば、置換する対象の符号化データ、又は、置換する対象の符号化データから計算されるCRC（「巡回冗長度検査」）である。例として、検査データは、置換する対象の符号化データのバイト全てのXOR（排他的論理和として知られている論理演算）である。検査データが、置換する対象の符号化データである場合、消費者向装置において、検査データが、置換する対象の受信符号化データと比較される。両方のデータが一致する場合、置換が行われ、さもなければ、置換は行われず、誤差が場合によっては報告される。検査データがCRCである場合、検査データが、置換する対象の受信符号化データに基づいて局所的に計算されたCRCと比較される。両方のデータが等しい場合、置換が行われ、さもなければ、置換は行われない。実際に、消費者向装置によって受信された符号化データのビットストリームにおいて、置換する対象の符号化データが、送信中に特定の修正を経た場合、置換は行われない。前述の修正は、画像変換、トランスコーディングによるものであり得る。CRCは、置換する対象の符号化データを挿入するために必要なビット数よりも少ないビットを使用して効果的に挿入することが可能である。

30

#### 【0036】

図4に示す別の変形によれば、3つ以上の置換データを工程12で挿入することが可能である。各置換データは、埋め込む対象のデータを表す。前述の場合、2つ以上のビットを一度に埋め込むことができる。図4に示す例として、4つの置換データが工程12で挿入される。置換データのうちの1つが符号化データ・ビットストリームに「00」を埋め込むために使用され、置換データのうちの第2のものが、符号化データ・ビットストリームに「11」を埋め込むために使用され、置換データの第2のものは、符号化データ・ビットストリームに「01」を埋め込むために使用され、置換データの第4のものは、符号化データ・ビットストリームに「10」を埋め込むために使用される。前述の変形は、検査データとの組合せで使用することができる。

40

#### 【0037】

第2の例示的な実施例は、SEIメッセージ（SEIは「Supplemental Enhancement Information」）を使用した、符号化標準のフレームワーク内を図1に関して説明する。前述のSEIメッセージは、Information

50

n technology - Coding of audio - visual objects - Part 10 : Advanced Video Codingと題するISO / IEC 1446の文献のAnnex Dに規定されている。H. 264に規定されたSEIメッセージは、前述の標準に決して限定されるものでない。前述のメッセージは、将来の標準によって使用することができ、H. 263などの標準によっても使用される。H. 263は、復号化処理の改変を必要とすることなく、復号化画像データの有用性を向上させることが可能ないくつかの特徴を含む(「SERIES H: AUDIOVISUAL AND MULTIMEDIA SYSTEMS - Infrastructure of audiovisual services - Coding of moving Video - Video coding for low bit rate communication」と題するITU-Tによる勧告のAnnex L及びWにおいて規定されている)。

10

#### 【0038】

一標準は、前述の標準と互換であるために、符号化データのストリームが順守しなければならない構文を規定する。構文は、特に、種々の情報項目(例えば、系列に含まれる画像に関するデータ、動きベクトル等)が符号化される。H. 264によれば、上記項目は、CABAC(「コンテキストベースの適応的二進算術符号化」の頭字語)を使用してエントロピ符号化される。復号化装置がその使用に必要な機能を有しない場合、前述のSEIは無視される。効果的には、この第2の例示的な実施例は、標準を修正することなく、透かし支援データを送信することが可能である。

20

工程12では、少なくとも1つの識別データ及び少なくとも1つの置換データが挿入される。

#### 【0039】

工程14では、フォーマット・データが、SEIメッセージを使用して符号化データFのビットストリームに挿入される。

この目的で、(透かし支援データをカプセル化するWSM SEIメッセージと呼ばれる新たなSEIメッセージが規定される。新たな構文が、ISO / IEC 14496 - 10という文献と同じ規則で、擬似符号の形式でアレイにおいて表す。特に、演算子「==」は「等しい」ということを表し、「!=」は「等しくない」ということを意味する。演算子「!」は、論理演算子「NOT」である。SEIメッセージは、「未登録データ」SEIペイロード・タイプ(SEI\_payloadTypeは5に等しい)を使用し、ISO / IEC 14496 - 10のsection 7.3.2.3.1を参照して以下の通りに規定される。

30

#### 【0040】



【表 1】

WSM_sei_message() {	記述子
<b>user_data_unregistered_payload_type_byte</b> /* equal to 0x05 */	u(8)
SEI_PayLoadSize = 0	
while( next_bits( 8 ) == 0xFF ) {	
<b>ff_byte</b> /* equal to 0xFF */	f(8)
SEI_PayLoadSize += 255	
}	
<b>last_payload_size_byte</b>	u(8)
SEI_PayLoadSize += last_payload_size_byte	
<b>WSM_user_data_unregistered</b> (SEI_PayLoadSize)	b(v)
}	

10

ここで、

20

f ( n ) は、左ビットを最初に、( 左から右に ) 書き込まれた n 個のビットを使用した固定パターン・ビット列であり、

u ( n ) は、n ビットを使用した、符号なし整数であり、

b ( 8 ) は、何れかのビット列 ( 8 ビット ) のパターンを有するバイトであり、

b ( v ) は可変長ビット・フィールドであり、

next\_bits ( n ) は、ビットストリーム・ポインタを進めることなく、比較目的で符号化データ・ビットストリームに、次のビットを供給する。したがって、それにより、符号化データのビットストリームにおける次の n ビットを見ることができ、n はその引数である。

【 0 0 4 1 】

30

ff\_byte は、使用される構文構造のより長い表現に対する必要性を表す 0 x F F に等しいバイトである。

【 0 0 4 2 】

user\_data\_unregistered\_payload\_type\_byte は、WSM SEI メッセージのペイロード・タイプである。例として、MPEG-4 AVC user\_data\_unregistered\_payload\_type\_byte は 0 x 0 5 に等しい。

【 0 0 4 3 】

last\_payload\_size\_byte は、SEI メッセージのサイズの最後のバイトである。

40

【 0 0 4 4 】

前述のタイプの SEI メッセージは、ISO / IEC 14496 - 10 という文献の section 7 . 3 . 2 . 3 . 1 において更に規定される。

【 0 0 4 5 】

WSM\_user\_data\_unregistered ( ) は以下のように規定される。

【 0 0 4 6 】

【表 2】

WSM_user_data_unregistered( SEI_PayLoadSize ) {	記述子
<b>uuid_iso_iec_11578</b>	u(128)
<b>WSM_Payload</b> (SEI_PayLoadSize – 16)	u(v)
}	

u u i d \_ i s o \_ i e c \_ 1 1 5 7 8 は、I S O / I E C 1 1 5 7 8 : 1 9 9 6 A n n e x A という文献に規定された手順により、U U I D ( U U I D は、汎用一義識別子を意味する ) として規定された値を有する。I E T F R F C 4 1 2 2 を使用して生成される前述の U U I D の例には、1 d f a 5 2 e 0 - e f 8 b - 1 1 d d - b a 2 f - 0 8 0 0 2 0 0 c 9 a 6 6 がある。

10

【 0 0 4 7 】

透かし支援データに関する情報を含む W S M \_ p a y l o a d の構文は、

【表 3】

WSM_Payload (WSM_PayloadSize) {	記述子	
<b>WSM_FormatType</b>	u(8)	
<b>WSM_NumberOfEntries</b>	u(4)	
<b>WSM_SequenceTag</b>	u(12)	
PayloadIndex = 2		
if (WSM_FormatType == FT0) {		10
Skip()		
}		
else if (WSM_FormatType == FT1) {		
for (i = 0 ; i ≤ WSM_NumberOfEntries ; i++) {		
<b>WSM_StringLength [i]</b>	u(4)	
<b>WSM_StringOffset [i]</b>	u(20)	
PayloadIndex += (20+4)/8		20
for (j = 0 ; j ≤ WSM_StringLength[i] ; j++) {		
<b>WSM_CurrentValue [i][j]</b>	u(8)	
}		
for (j = 0 ; j ≤ WSM_StringLength[i] ; j++) {		
<b>WSM_OtherValue [i][j]</b>	u(8)	
}		
PayloadIndex += 2 * (WSM_StringLength[i] + 1)		
}		30
}		
else if (WSM_FormatType == FT2) {		
for (i = 0 ; i ≤ WSM_NumberOfEntries ; i++) {		
<b>WSM_StringLength [i]</b>	u(4)	
<b>WSM_StringOffset [i]</b>	u(20)	
PayloadIndex += (20+4)/8		
for (j = 0 ; j ≤ WSM_StringLength[i] ; j++) {		40

<b>WSM_CurrentValue [i][j]</b>	u(8)	
}		
for (j = 0 ; j ≤ WSM_StringLength[i] ; j++) {		
<b>WSM_OneValue [i][j]</b>	u(8)	
}		
for (j = 0 ; j ≤ WSM_StringLength[i] ; j++) {		
<b>WSM_ZeroValue [i][j]</b>	u(8)	10
}		
PayLoadIndex += 3 * (WSM_StringLength[i] + 1)		
}		
}		
else if (WSM_FormatType == FT3) {		
for (i = 0 ; i ≤ WSM_NumberOfEntries ; i++) {		
<b>WSM_StringLength [i]</b>	u(4)	20
<b>WSM_StringOffset [i]</b>	u(20)	
<b>WSM_CurrentValueCRC [i]</b>	u(8)	
PayLoadIndex += (20+4)/8 + 1		
for (j = 0 ; j ≤ WSM_StringLength[i] ; j++) {		
<b>WSM_OtherValue [i][j]</b>	u(8)	
}		
PayLoadIndex += (WSM_StringLength[i] + 1)		
}		30
}		
else if (WSM_FormatType == FT4) {		
for (i = 0 ; i ≤ WSM_NumberOfEntries ; i++) {		
<b>WSM_StringLength [i]</b>	u(4)	
<b>WSM_StringOffset [i]</b>	u(20)	
<b>WSM_CurrentValueCRC [i]</b>	u(8)	
PayLoadIndex += (20+4)/8 + 1		
for (j = 0 ; j ≤ WSM_StringLength[i] ; j++) {		40

<b>WSM_OneValue [i][j]</b>	u(8)	
}		
for (j = 0 ; j ≤ WSM_StringLength[i] ; j++) {		
<b>WSM_ZeroValue [i][j]</b>	u(8)	
}		
PayLoadIndex += 2 * (WSM_StringLength[i] + 1)		
}		10
}		
else if (WSM_FormatType == FT5) {		
for (i = 0 ; i ≤ WSM_NumberOfEntries ; i++) {		
<b>WSM_StringLength [i]</b>	u(4)	
<b>WSM_PacketOffset [i]</b>	u(12)	
<b>WSM_StringRelativeOffset [i]</b>	u(8)	
PayLoadIndex += (20+4)/8		20
for (j = 0 ; j ≤ WSM_StringLength[i] ; j++) {		
<b>WSM_CurrentValue [i][j]</b>	u(8)	
}		
for (j = 0 ; j ≤ WSM_StringLength[i] ; j++) {		
<b>WSM_OtherValue [i][j]</b>	u(8)	
}		
PayLoadIndex += 2 * (WSM_StringLength[i] + 1)		
}		30
}		
else if (WSM_FormatType == FT6) {		
for (i = 0 ; i ≤ WSM_NumberOfEntries ; i++) {		
<b>WSM_StringLength [i]</b>	u(4)	
<b>WSM_PacketOffset [i]</b>	u(12)	
<b>WSM_StringRelativeOffset [i]</b>	u(8)	
PayLoadIndex += (20+4)/8		40
for (j = 0 ; j ≤ WSM_StringLength[i] ; j++) {		

<b>WSM_CurrentValue [i][j]</b>	u(8)	
}		
for (j = 0 ; j ≤ WSM_StringLength[i] ; j++) {		
<b>WSM_OneValue [i][j]</b>	u(8)	
}		
for (j = 0 ; j ≤ WSM_StringLength[i] ; j++) {		
<b>WSM_ZeroValue [i][j]</b>	u(8)	10
}		
PayLoadIndex += 3 * (WSM_StringLength[i] + 1)		
}		
}		
else if (WSM_FormatType == FT7) {		
for (i = 0 ; i ≤ WSM_NumberOfEntries ; i++) {		
<b>WSM_StringLength [i]</b>	u(4)	20
<b>WSM_PacketOffset [i]</b>	u(12)	
<b>WSM_StringRelativeOffset [i]</b>	u(8)	
<b>WSM_CurrentValueCRC [i]</b>	u(8)	
PayLoadIndex += (20+4)/8 + 1		
for (j = 0 ; j ≤ WSM_StringLength[i] ; j++) {		
<b>WSM_OtherValue [i][j]</b>	u(8)	
}		
PayLoadIndex += (WSM_StringLength[i] + 1 )		30
}		
}		
else if (WSM_FormatType == FT8) {		
for (i = 0 ; i ≤ WSM_NumberOfEntries ; i++) {		
<b>WSM_StringLength [i]</b>	u(4)	
<b>WSM_PacketOffset [i]</b>	u(12)	
<b>WSM_StringRelativeOffset [i]</b>	u(8)	
<b>WSM_CurrentValueCRC [i]</b>	u(8)	40

PayLoadIndex += (20+4)/8 + 1	
for (j = 0 ; j ≤ WSM_StringLength[i] ; j++) {	
WSM_OneValue [i][j]	u(8)
}	
for (j = 0 ; j ≤ WSM_StringLength[i] ; j++) {	
WSM_ZeroValue [i][j]	u(8)
}	
PayLoadIndex += 2 * (WSM_StringLength[i] + 1 )	
}	
}	
else {	
Unsupported ()	
}	
for (i = PayLoadIndex ; i < WSM_PayLoadSize ; i++) {	
WSM_Filler [i]	u(8)
}	
}	

10

20

として規定される。

【 0 0 4 8 】

列又はC A B A C列は、一連のビットとして規定される。したがって、以下では、「列」の語は、置換される符号化データ・ビットストリームにおける連続ビットを表す。

WSM\_FormatTypeは、置換が行われるやり方、及び、よって、透かし支援データ・エントリのフォーマットを規定するために使用されるバイトである。以下のフォーマットは、例として規定される。

30

【 0 0 4 9 】

FT0；エントリは空白であり、誤りを報告することはできない。

【 0 0 5 0 】

FT1：一置換値WSM\_OtherValueのみが「1」を埋め込むことが提案されている一方、現在の値WSM\_CurrentValueは「0」を埋め込むために使用される。現在値WSM\_CurrentValueは、置換しなければならないかもしれないC A B A C符号化タイプである。現在の値（WSM\_CurrentValue）も、個人化処理のロバスト性の増加を促進するための検査データとして送信される。

40

【 0 0 5 1 】

FT2：2つの置換値が提案されている。WSM\_ZeroValueは「0」を埋め込む一方、WSM\_OneValueは「1」を埋め込む。現在の値（WSM\_CurrentValue）も、個人化処理のロバスト性の増加を促進するための検査データとして送信される。

【 0 0 5 2 】

FT3：FT1と同様。ビットレートを節減するために、WSM\_CurrentValue列を、単純な1バイトCRCで置き換える。前述のWSM\_CurrentValueCRCは、個人化処理をよりロバストにするために、消費者向装置において検査データとして使用される。

50

## 【 0 0 5 3 】

FT4 : FT2と同様。ビットレートを節減するために、WSM\_CurrentValue列を、単純な1バイトCRCで置き換える。前述のWSM\_CurrentValueCRCは、個人化処理をよりロバストにするために、消費者向装置において検査データとして使用される。

## 【 0 0 5 4 】

FT5 : FT1と同様であるが、WSM\_StringOffsetは、向けられたパケット内のバイト・アドレス(WSM\_StringRelativeOffset)、及びパケット・アドレス(WSM\_PacketOffsetという2つのサブフィールドに分けられる。

10

## 【 0 0 5 5 】

FT6 : FT2と同様であるが、WSM\_StringOffsetは、向けられたパケット内のバイト・アドレス(WSM\_StringRelativeOffset)、及びパケット・アドレス(WSM\_PacketOffset)という2つのサブフィールドに分けられる。

## 【 0 0 5 6 】

FT7 : FT3と同様であるが、WSM\_StringOffsetは、向けられたパケット内のバイト・アドレス(WSM\_StringRelativeOffset)、及びパケット・アドレス(WSM\_PacketOffset)という2つのサブフィールドに分けられる。

20

## 【 0 0 5 7 】

FT8 : FT4と同様であるが、WSM\_StringOffsetは、向けられたパケット内のバイト・アドレス(WSM\_StringRelativeOffset)、及びパケット・アドレス(WSM\_PacketOffset)という2つのサブフィールドに分けられる。

## 【 0 0 5 8 】

その他：将来の使用のために予約されている。誤りが報告される。

## 【 0 0 5 9 】

例として、FT0 = 0x00、FT1 = 0x01、FT2 = 0x02、FT3 = 0x03、FT4 = 0x04、FT5 = 0x05、FT6 = 0x06、FT7 = 0x07及びFT8 = 0x08である。当然、他の値を使用することができる。

30

## 【 0 0 6 0 】

フォーマットFT1乃至FT8の場合、検査データは、個人化処理をよりロバストにするために、工程12で、ストリームにおいて更に符号化される。消費者向装置では、WSM\_StringOffsetによって向けられるCABACバイト列を置換する前に、前述のCABAC列がWSM\_CurrentValueと比較される。一致しない場合、置換は行われず、誤りが場合によっては報告される。変形によれば、CRCが、WSM\_StringOffsetによって向けられるCABACバイト列から局所的に計算され、WSM\_CurrentValueCRCと比較される。一致しない場合、置換は行われず、誤りが場合によっては報告される。より多くのWSM\_FormatTypeを導入して、より多くのビット(例えば、4でなく8)がWSM\_NumberOfEntriesを符号化し、WSM\_StringLengthに対する24ビットのアドレス(又はWSM\_PacketOffsetに対する16ビット)を可能にし得る。

40

第1の例として、FT2に等しい新たなフォーマット・タイプWSM\_FormatType = FT9が規定される(すなわち、WSM\_CurrentValueに加えて2つの置換値)。FT9フォーマット・タイプは、WSM\_NumberOfEntriesが(4でなく)8ビットに対して符号化され、WSM\_StringLengthが(4でなく)24ビットに対して符号化されるという点でFT2と異なる。これは、より大きなアクセス・ユニットを扱うことを可能にする。前述のフォーマットは、高ビット・レート符号化によく適している。

50



第2の例として、別のフォーマットFTnが規定され、nは、既に使用されたものと異なる正の整数である。前述のフォーマットによれば、置換エントリは全て、同じ列長を有する。したがって、透かし支援データ・ペイロード容量を最小にするために、WSM\_StringLength[i]は、全てのエントリに共通にされ、よって、ループ「for (i = 0 ; i WSM\_NumberOfEntries ; i++)」から除去され、WSM\_Payloadヘッダ（例えば、WSM\_SequenceTagの直後）に移動させる。WSM\_StringLength[i]はよって、大局的なWSM\_StringLengthに変更される。

#### 【0061】

WSM\_NumberOfEntriesは、現在の透かし支援データにおいて存在している、エントリ数 - 1を規定する。置換は更に、エントリ毎に詳説される。

10

#### 【0062】

変形によれば、WSM\_NumberOfEntriesは、エントリ数を規定する。

WSM\_SequenceTagは、透かし支援データの第1のエントリに、デジタル符号ペイロードのどのビット（すなわち、第1のビット、第2のビット、第3のビット等）を挿入すべきかを示す。ペイロードは「ユーザID」を含み、「タイム・スタンプ」、他の情報（例えば、著作権）、保護ビット、冗長度を更に含み得る。

#### 【0063】

WSM\_StringLengthは、置換列の長さ - 1を（例えば、バイト数で）規定する。

20

変形によれば、WSM\_StringLengthは、置換列の長さを規定する。

WSM\_StringOffsetは、現在のアクセス・ユニットにおいて置換される対象の第1のバイトのバイト数での絶対アドレスである。WSM\_StringOffsetは、0に等しい場合、現在のアクセス・ユニットの第1のスライスの第1のバイトを識別する。アクセス・ユニットは、「ちょうど1つのプライマリ符号化ピクチャを常に含むNALユニットの組」として規定される。NALユニットは、「あとに続くデータのタイプ及びそのデータを含むバイトの表示を含む構文構造」である。アクセス・ユニット及びNALユニットは、MPEG-4 AVC/H.264標準の一部である。

#### 【0064】

特定の特徴によれば、エントリは、オフセット昇順で編成される。

30

#### 【0065】

変形によれば、WSM\_StringOffsetは2つのサブフィールドに分けられ、第1のサブフィールドWSM\_PacketOffsetはTSパケット・アドレスであり、第2のサブフィールドWSM\_StringRelativeOffsetは、TSパケットの先頭からの、バイト数でのオフセット値である。これにより、消費者向装置における処理が緩和され得る。この場合、以下のフィールドが符号化される。すなわち、

WSM\_PacketOffsetは、0に等しい場合、WSM\_Payloadの第1のバイトを含むTSパケットを識別する。

WSM\_StringRelativeOffsetは、WSM\_PacketOffsetによって向けられるTSパケットにおいて置換される対象の第1のバイトの相対アドレスである。WSM\_StringRelativeOffsetは、0に等しい場合、TSパケット・ペイロードの第1バイトを識別する。

40

WSM\_Fillerは、フィラー・バイトである。例として、WSM\_Filler[i]は、0xFFに等しい。

#### 【0066】

WSM\_Fillerは、WSM\_PayloadSizeが大きすぎる場合、スタッフィングに使用することが可能である。

#### 【0067】

H.264では、前述の私的SEIメッセージは、画像に対する正確な時間レファレン

50

スを提供するNALユニット（「ネットワーク適合層」の頭字語）を使用して符号化データ・ビットストリームに埋め込まれ、よって、透かし支援データと画像符号化データとの間の同期化を促進する。実際に、SEIメッセージは、アクセス・ユニットに「添付」される。MPEG4/AVC符号化モード（フィールド、フレーム、MBAFF）に応じて、アクセス・ユニットは1つの完全なフィールド又は1つの完全なフレームを含む。WSM SEIメッセージはしたがって、以下のようにNALユニットにおいてカプセル化される。

【0068】

【表4】

nal_unit {	記述子
<b>forbidden_zero_bit</b> /* shall be 0 */	f(1)
<b>nal_ref_idc</b> /* shall be 0 for a SEI message */	u(2)
<b>nal_unit_type</b> /* equal to 0x06 for a SEI message */	u(5)
<b>Other SEI messages()</b> /* Optional */	b(v)
<b>WSM_sei_message()</b>	b(v)
<b>Other SEI messages()</b> /* Optional */	b(v)
<b>rbsp_trailing_bits()</b>	
}	

**forbidden\_zero\_bit**は、H.264により、ゼロに等しい。

【0069】

**nal\_ref\_idc**は、H.264により、6、9、10、11又は12に等しい**nal\_unit\_type**を有するNALユニット全てについて0に等しい。

**nal\_unit\_type**は、ISO/IEC14496-10の表7-1に規定されたNALユニットに含まれるRbsp（「未処理バイト系列ペイロード」を意味する）のタイプを規定する。VCL NALユニット（VCLは、「ビデオ符号化層」の頭字語である）は、1乃至5に等しい**nal\_unit\_type**を有するNALユニットとして規定される。残りのNALユニットは全て、非VCL NALユニットと呼ばれる。6に等しい**Nal\_unit\_type**はSEIメッセージを意味する。

【0070】

**rbsp\_trailing\_bits()**は、特定の可変長符号のビット列である。

実際には、一般に、系列0x80が戻され、次に0x00が戻される。

【0071】

第3の例示的な実施例を、SEIメッセージを使用したH.264/MPEG4 AVC標準のフレームワーク内で提案する。前述の第3の例示的な実施例では、新たなタイプの情報SEIが、WSMに関する更なる情報を符号化するように規定される。この目的で、フィールド**payloadType**に対する新たな値が、未使用値のうちから規定される（例えば、**payloadType**が22に等しい）。特に、（0乃至21の）**payloadType**の最初の22の値は、例えば、19に等しい**payloadType**に対応するフィルムのグレインの特徴などの特定の情報を符号化するために既に使用されている。新たな構文を、ISO/IEC14496-10という文献中の規則と同じ規則で、擬似符号のフォーマットでアレイ内に表す。SEIデータの構文（すなわち、**sei\_payload**）は、

【表 5】

sei_payload( payloadType, payloadSize ) {	C	記述子
if( payloadType == 0 )		
buffering_period( payloadSize )	5	
.....		
else if( payloadType == 21 )		
stereo_video_info( payloadSize )	5	
else if ( payloadType == 22 )		
<b>WSM_payload(WSM_payloadSize)</b>	5	
Else		
reserved_sei_message( payloadSize )	5	
if( !byte_aligned( ) ) {		
bit_equal_to_one /* equal to 1 */	5	f(1)
while( !byte_aligned( ) )		
bit_equal_to_zero /* equal to 0 */	5	f(1)
}		
}		

として拡張され、WSM\_payloadは、第2の実施例のWSM\_payloadと同一である。

【0072】

byte\_aligned( )は、符号化データのビットストリームがバイト境界上でアラインされているか否かをチェックするために使用される。

【0073】

本発明は、図5及び図6を参照して説明した、透かしを入れる方法に関する。より詳細には、本発明は、フォーマット・データ、識別データ及び置換データを含む符号化データのビットストリームF+WSMに透かしを入れる方法に関する。

【0074】

工程20では、置換データによって置換される対象の符号化データを置換するやり方を規定するフォーマット・データ、置換する対象の符号化データを識別する識別データ、及び置換する対象の符号化データ毎の少なくとも1つの置換データは、符号化データFのビットストリームの一部を復号化することによって求められる。例として、第2の例示的な実施例及び第3の例示的な実施例に対応して、WSMをカプセル化するSEIメッセージが復号化される。

【0075】

工程22では、置換される対象の符号化データが識別データによって識別される。

【0076】

工程24では、符号化データFのビットストリームは、適切な置換データによって置換される、識別された符号化データそれぞれを置換することにより、透かしを入れたビットストリームF'に透かしが入れられる。適切な置換データは、置換データに基づいて、更に、符号化データFのビットストリームにおいて埋め込む対象の透かしWの値に基づいて、フォーマット・データに応じて求められる。

【0077】

例として、図6では、置換される対象の3つの符号化データC1、C2及びC3は、工

10

20

30

40

50

程 2 0 において回復される識別データから工程 2 2 で識別される。

工程 2 4 では、前述の 3 つの符号化データそれぞれが、埋め込む対象のデジタル符号 W に応じて置換される。例として、W は 0 0 1 に等しい。したがって、ビット 0 を C 1 に埋め込むために、C 1 が E 0 で置換される一方、E 1 はビット 1 を埋め込むために使用される。次いで、ビット 0 を C 2 に埋め込むために、C 2 が F 0 で置換される一方、F 1 はビット 1 を埋め込むために使用される。最後に、ビット 1 を C 3 に埋め込むために、C 3 は G 1 で置き換えられる一方、G 0 は、ビット 0 を埋め込むために使用される。

【 0 0 7 8 】

変形によれば、工程 2 0 中に、検査データが求められ、置換が行われる（工程 2 4。検査データが、置換される対象の受信データに一致する場合にのみ）。

10

【 0 0 7 9 】

本発明は、画像、オーディオ・データ、3 D データ、又は何れかの種類のデジタル・コンテンツの系列を表す符号化データのビットストリーム F に更に関する。本発明による符号化データのビットストリームは、透かし支援データを含み、特に、

置換する対象の符号化データを識別する識別データ、

置換する対象の符号化データに対する少なくとも 1 つの置換データ、及び

置換データによって置換する対象の符号化データを置換するやり方を規定するフォーマット・データを含む。

【 0 0 8 0 】

本発明は、図 7 を参照して説明した装置 3 に更に関する。装置 3 は、画像、又は透かし支援データを含む何れかのデジタル・コンテンツの系列を表す符号化データのビットストリーム F を受信するためのエントリ 3 0 を含む。符号化データ F のビットストリームは効果的には、3 つの例示的な実施例のうちの 1 つによる、挿入の方法によって生成される。エントリ 3 0 の出力は、透かし入れモジュール 3 2 の入力に接続される。透かし入れモジュール 3 2 の出力は、装置 3 の第 1 の出力 3 6 及び復号化モジュール 3 4 の入力に接続される。復号化モジュール 3 4 の出力は、装置 3 の第 2 の出力 3 8 に接続される。

20

【 0 0 8 1 】

透かし入れモジュール 3 2 は、本発明による透かし入れ方法の工程 2 0、2 2 及び 2 4 を実現するよう適合される。この目的で、透かし入れモジュール 3 2 は、

置換データによって置換する対象の符号化データを置換するやり方を規定するフォーマット・データ、置換する対象の符号化データ毎の少なくとも 1 つの置換データ、及び置換する対象の符号化データを識別する識別データを符号化データ・ビットストリームから判定する装置と、

30

識別データにより、置換する対象の符号化データを識別する装置と、

フォーマット・データ及び透かしデータに基づいて、1 つの置換符号化データで置換される対象の画像符号化データを置換することにより、符号化データ・ビットストリームに透かしを入れる透かし入れ装置とを備える。

【 0 0 8 2 】

F に埋め込む対象の透かし W は、少なくとも、装置 3 を識別することを可能にするユーザ ID / 識別子を含む。第 1 の出力 3 6 は、WLAN（「無線ローカル・エリア・ネットワーク」の頭字語）などのネットワーク R に送信することができる透かしを入れたビットストリーム F' を出力する。復号化モジュール 3 4 は、透かしを入れた画像系列などの、透かしを入れたデジタル・コンテンツ S' を再構成するために、透かしを入れたストリーム F' を復号化するよう適合される。S' は表示するために TV セットに送信することができる。

40

【 0 0 8 3 】

効果的な実施例によれば、バッファ複製、及び、よって、侵害者による、透かし入れ支援データの回復を避けるために、付加された後に符号化データ・ビットストリームにおいて、透かし入れ支援データが消去される。

【 0 0 8 4 】

50

符号化手法の第2の例示的な実施例に結びつけられた例として、以下の処理を、透かし入れ支援データを消去するよう適用することができる。

【0085】

WSM\_\_FormatType = 0 にセットする。

WSM\_\_NumberOfEntries = 0 にセットする。

【0086】

ペイロードの残りをWSM\_\_Fillerで埋める。

【0087】

SEIメッセージは、画像に対する正確な時間レファレンスを提供するNALユニット（「ネットワーク適合層」の頭字語）を使用して符号化データ・ビットストリームに埋め込まれ、よって、透かし支援データと画像符号化データとの間の同期化を促進する。更に、SEIメッセージは、圧縮画像系列と同じ暗号化システムで暗号化され、深くマージされる。よって、透かし入れ支援データを除去するために、暗号化システムを打ち破ることが必要である。最後に、SEIメッセージは、ペイロードの点では制限されない。

10

【0088】

当然、本発明は上記実施例に限定されない。より詳細には、本発明は、画像符号化データ・ビットストリームへのデータの挿入に限定されない。実際に、データが挿入された符号化データ・ビットストリームは、例えば、オーディオ又は3Dデータなどの何れかの種類のデジタル・コンテンツを表し得る。

【0089】

更に、本発明は、MPEG-4 AVC ビデオ符号化標準に決して限定されるものではない。

20

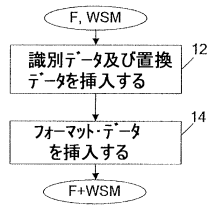
【符号の説明】

【0090】

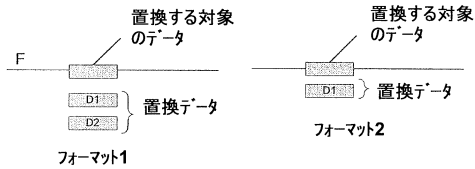
12 工程

14 工程

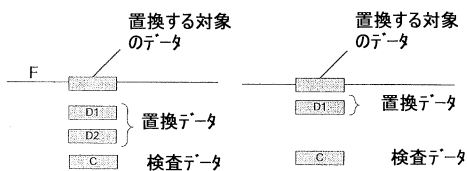
【図 1】



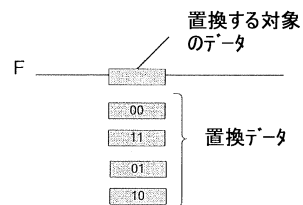
【図 2】



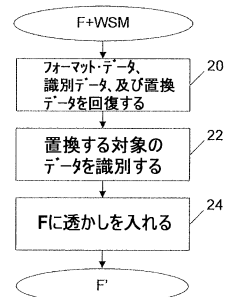
【図 3】



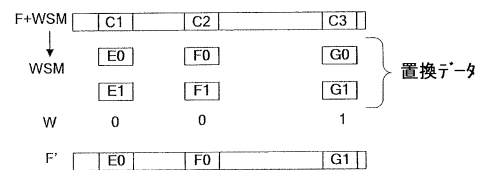
【図 4】



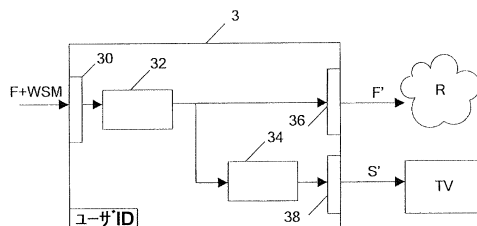
【図 5】



【図 6】



【図 7】



---

フロントページの続き

(72)発明者 マリー ジャン コレティ

フランス国 3 5 5 1 0 セゾン・セヴィニエ リュ・ド・パン 1 ビス

(72)発明者 パスカル マリー

フランス国 3 5 7 4 0 パース アヴェニュー・タル・コート 1 6

審査官 坂東 大五郎

(56)参考文献 特表2009-518945(JP,A)

(58)調査した分野(Int.Cl., DB名)

H04N 19/00 - 19/98

H04N 7/025 - 7/088