

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号

特許第3597647号  
(P3597647)

(45) 発行日 平成16年12月8日(2004.12.8)

(24) 登録日 平成16年9月17日(2004.9.17)

(51) Int.Cl.<sup>7</sup>

F I

H04L 1/00  
H04L 7/00  
H04N 7/24  
// H03M 7/30H04L 1/00 F  
H04L 7/00 D  
H04N 7/13 A  
H03M 7/30 Z

請求項の数 7 (全 42 頁)

(21) 出願番号 特願平8-243883  
 (22) 出願日 平成8年9月13日(1996.9.13)  
 (65) 公開番号 特開平10-126389  
 (43) 公開日 平成10年5月15日(1998.5.15)  
 審査請求日 平成15年9月16日(2003.9.16)  
 (31) 優先権主張番号 特願平7-276993  
 (32) 優先日 平成7年9月29日(1995.9.29)  
 (33) 優先権主張国 日本国(JP)  
 (31) 優先権主張番号 特願平8-61450  
 (32) 優先日 平成8年3月18日(1996.3.18)  
 (33) 優先権主張国 日本国(JP)  
 (31) 優先権主張番号 特願平8-163082  
 (32) 優先日 平成8年6月24日(1996.6.24)  
 (33) 優先権主張国 日本国(JP)

(73) 特許権者 000003078  
 株式会社東芝  
 東京都港区芝浦一丁目1番1号  
 (74) 代理人 100058479  
 弁理士 鈴江 武彦  
 (74) 代理人 100084618  
 弁理士 村松 貞男  
 (74) 代理人 100068814  
 弁理士 坪井 淳  
 (74) 代理人 100092196  
 弁理士 橋本 良郎  
 (74) 代理人 100091351  
 弁理士 河野 哲  
 (74) 代理人 100088683  
 弁理士 中村 誠

最終頁に続く

(54) 【発明の名称】 符号化方法及び装置

(57) 【特許請求の範囲】

【請求項1】

画像信号を圧縮符号化して生成される複数種類の可変長符号を多重化して多重化符号列を生成し、前記多重化符号列を入力として出力符号列を組み立て、該出力符号列中の予め周期的に定められた複数の同期符号挿入位置のいずれかに同期符号を挿入することを特徴とする符号化方法。

【請求項2】

画像信号を可変長符号化して生成される複数種類の可変長符号を多重化して多重化符号列を生成する多重化手段と、  
 前記多重化符号列を入力として出力符号列を組み立てる符号列組立手段とを有し、  
 前記符号列組立手段は、前記出力符号列中の予め周期的に定められた複数の同期符号挿入位置のいずれかに同期符号を挿入することを特徴とする符号化装置。

10

【請求項3】

前記多重化手段は、前記可変長符号を前記画像信号のフレーム単位で多重化することを特徴とする請求項2に記載の符号化装置。

【請求項4】

前記多重化手段は、前記可変長符号を前記画像信号のフレームの部分領域単位で多重化することを特徴とする請求項2に記載の符号化装置。

【請求項5】

前記多重化手段は、前記可変長符号を前記画像信号のフレーム単位で多重化し、

20

前記符号列組立手段は、前記多重化符号列の前記フレーム単位で多重化された各多重化単位の終りの部分の直前または直後に位置する同期符号挿入位置に同期符号を挿入することを特徴とする請求項 2 に記載の符号化装置。

【請求項 6】

前記多重化手段は、前記可変長符号を前記画像信号のフレームの部分領域単位で多重化し、

前記符号列組立手段は、前記多重化符号列の前記部分領域単位で多重化された各多重化単位の終りの部分の直前または直後に位置する同期符号挿入位置に同期符号を挿入することを特徴とする請求項 2 に記載の符号化装置。

【請求項 7】

前記出力符号列中の前記同期符号挿入位置にある前記同期符号以外の符号列を該同期符号とのハミング距離が予め定められた値以上となるように変換する符号列変換手段をさらに有することを特徴とする請求項 2 乃至 6 のいずれか 1 項に記載の符号化装置。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、無線伝送路のような誤り率の高い媒体を介して情報を伝送 / 蓄積するシステムに係り、高能率圧縮符号化によって得られた圧縮符号列を誤り訂正 / 検出符号化して伝送 / 蓄積するのに適した符号化方法及び装置に関する。

【0002】

【従来の技術】

例えば、無線 TV 電話、携帯情報端末、デジタル TV 放送システムのように、画像や音声の情報をできるだけ少ない情報量となるように高能率に圧縮符号化して無線伝送路等を介して伝送するシステムにおいては、伝送路の誤り率が高いため、得られた符号列をいかに高品質に伝送するかが重要な課題となる。

【0003】

このように誤り率の高い媒体を介して符号列の伝送 / 蓄積を行う場合に、誤り率を低減する手段として、BCH 符号、RS 符号、畳み込み等の誤り訂正符号が多く用いられる。また、受信側で誤り検出を可能にする手段としては、チェックサムやCRC等の誤り検出符号が用いられる。これらの誤り訂正 / 誤り検出は、伝送 / 蓄積する情報に余分なビット（冗長性）を一定の規則に従って付加し、復号の際に伝送 / 蓄積された符号列がこの規則に従っているかどうかを調べ、その結果によって誤りの訂正や検出を行うものである。

【0004】

しかし、このように高能率圧縮符号化により得られた符号列を誤り訂正 / 検出符号に符号化して伝送 / 蓄積する方法は、伝送路 / 媒体での符号語誤りによって生じる同期外れを回復する同期回復手法との組み合わせが難しくなることが欠点である。同期回復手法としては、一意に復号可能な同期符号と呼ばれる符号を挿入し、同期外れが生じたときは同期符号を検出した時点から復号を再開する方法が多く用いられている。

【0005】

同期符号を一意に復号可能な符号語とするためには、他の符号語の組み合わせで同期符号と同一のビットパターンが生じないように符号語を構成する必要がある。しかし、一般に誤り訂正 / 検出符号化では、ある特定のビットパターンの出現を避けるように符号語を構成することは困難であり、同期符号と同一ビットパターンが出現した場合には、同期符号の誤検出により擬似同期を引き起こしてしまうことになる。

【0006】

この問題を避けるために、誤り訂正 / 検出符号化を行った後に同期符号と同一のビットパターンが符号列中に存在するか否かを判定し、同一ビットパターンが存在する場合には、そのパターン中にある規則に従ってダミービットを挿入し、復号化装置において同一の規則でダミービットを削除することにより擬似同期を防ぐ方法が用いられている。しかし、誤りが生じやすい媒体を介して符号列を伝送 / 蓄積する場合には、この挿入ビットにも誤

10

20

30

40

50

りが生じる可能性があり、新たな同期外れや擬似同期を引き起こすという問題点がある。

【 0 0 0 7 】

また、符号列を誤り訂正 / 検出符号化し同期符号を挿入する場合、従来では同期符号と同期符号とで挟まれた同期区間の最後の部分で、誤り訂正 / 検出符号化すべき情報ビットの剰余を補うために多くの挿入ビットを符号列に付加する必要があるため、符号化効率が低下するという問題点もある。

【 0 0 0 8 】

一方、誤り訂正 / 検出能力を高めるためには、伝送 / 蓄積する情報の冗長度を高めればよいが、こうすると同じ情報を送るのに必要なビット数は増える。このため、誤り訂正 / 検出能力を高めると、より伝送レートの高い伝送路が必要となったり、蓄積しなければならない情報のビット数が増加することになる。また、伝送レートや蓄積容量が同一ならば、冗長度が高いほど少ない情報しか伝送 / 蓄積できないことになる。画像や音声を高能率に圧縮符号化して伝送 / 蓄積する場合、誤り耐性を向上させるために冗長性を付加することは、伝送 / 蓄積レートが同一ならばより少ない情報量に圧縮符号化を行わなければならないことになり、画質や音質の低下を招くことになる。

10

【 0 0 0 9 】

そこで、より少ない冗長度で高い誤り耐性を持たせる方法として、階層符号化と呼ばれる方法がある。これは高能率圧縮符号化された情報を誤りが画質や音質に与える大きさに応じて分類し、誤りの影響が大きい情報には冗長度が高くともより誤り訂正 / 検出能力の高い誤り訂正 / 検出符号を用い、誤りの影響があまり大きくない情報には誤り訂正 / 検出能力はそれほど高くないが冗長度が少ない誤り訂正 / 検出符号を用いることにより、全体に同一の誤り訂正 / 検出符号を用いる場合より同一の平均冗長度でより誤り耐性を高める方法である。

20

【 0 0 1 0 】

例えば、動画像の高能率圧縮符号化に多く用いられている動き補償予測と直交変換を組み合わせた符号化方式、すなわち、入力される動画像信号に対して動き補償予測を行い、その予測残差を D C T ( 離散コサイン変換 ) などにより直交変換する方式においては、誤りが生じると大きな画質劣化を生じる動きベクトル情報や、予測残差信号の直交変換係数のうちの低次係数には、誤り訂正 / 検出能力の強い誤り訂正 / 検出符号を用い、誤りの影響が小さい予測残差信号の直交変換係数のうちの高次係数には、誤り訂正 / 検出能力の弱い誤り訂正 / 検出符号を用いる。

30

【 0 0 1 1 】

このような階層符号化を実現するためには、出力される符号列の途中で、誤り訂正 / 検出能力の異なる誤り訂正 / 検出符号を切り替える必要がある。誤り訂正 / 検出能力の異なる誤り訂正 / 検出符号を切り替える方法としては、誤り訂正 / 検出符号の種類を示すヘッダ情報を符号列に付加する方法がある。図 1 1 は、ヘッダ情報を付加して誤り訂正 / 検出符号を切り替えた符号列の例を示したものであり、この例では 2 種類の誤り訂正 / 検出符号 F E C 1 および F E C 2 を切り替えている。ヘッダ 1 1 0 1 ~ 1 1 0 4 には、誤り訂正 / 検出符号の種類や符号語の数を示すヘッダ情報が入っている。符号化装置では、これらのヘッダ情報に続けて誤り訂正 / 検出符号化された符号語を並べ、復号化装置ではヘッダ情報を復号し、これに従って誤り訂正 / 検出符号の復号を行う。

40

【 0 0 1 2 】

しかし、このようなヘッダ情報を付加することによって誤り訂正 / 検出符号を切り替える方法では、ヘッダ情報の付加によって伝送 / 蓄積しなければならない符号列のビット数が増加するという問題がある。画像や音声を高能率圧縮符号化して伝送 / 蓄積する場合、ヘッダ情報にビット数が割かれることは、画像や音声の高能率圧縮符号化に用いられるビット数が減少することになり、結果的に画質や音質の劣化を招く。

【 0 0 1 3 】

【 発明が解決しようとする課題 】

上述したように、動画像信号などを高能率圧縮符号化した符号列に対して誤り訂正 / 検出

50

符号化を行うと、任意のビットパターンが発生するため、誤り訂正 / 検出符号化と、一意に復号可能な同期符号を用いる同期回復手法との組み合わせた場合には、同期符号の誤検出による擬似同期という問題点があり、ダミービットの挿入により擬似同期を避ける操作を行った場合にも挿入ビットの誤りによって新たな同期外れや擬似同期を生じてしまうという問題点があった。

【 0 0 1 4 】

また、符号列を誤り訂正 / 検出符号化し同期符号を挿入する場合、従来では同期区間の最後の部分で、誤り訂正 / 検出符号化すべき情報ビットの剰余を補うために多くの挿入ビットを用いる必要があるため、符号化効率が低下するという問題点があった。

【 0 0 1 5 】

さらに、ヘッダ情報を付加することにより誤り訂正 / 検出能力の異なる誤り訂正 / 検出符号を切り替える符号化 / 復号化装置では、ヘッダ情報の付加により伝送 / 蓄積しなければならないビット数が増加するため、画像や音声を高能率圧縮符号化して伝送 / 蓄積する場合には、画像や音声の情報に割り当てられる情報量が減少して画質や音質の低下を招くという問題点があった。

【 0 0 1 6 】

本発明の主たる目的は、同期符号の誤検出による擬似同期や同期外れの問題を解決できる符号化方法及び装置を提供することにある。

【 0 0 2 0 】

【課題を解決するための手段】

上記の課題を解決するため、本発明に係る符号化方法は、画像信号を圧縮符号化して得られた複数種類の可変長符号を多重化して多重化符号列を生成し、多重化符号列を入力として出力符号列を組み立て、該出力符号列中の予め周期的に定められた複数の同期符号挿入位置のいずれかに同期符号を挿入することを特徴とする。

【 0 0 2 1 】

本発明に係る符号化装置は、画像信号を可変長符号化して得られた複数種類の可変長符号を多重化して多重化符号列を生成する多重化手段と、該多重化符号列を入力として出力符号列を組み立てる符号列組立手段とを有し、符号列組立手段は、出力符号列中の予め周期的に定められた複数の同期符号挿入位置のいずれかに同期符号を挿入することを特徴とする。

【 0 0 3 7 】

この場合、前記符号列組立手段は、前記出力符号列中の予め周期的に定められた複数の同期符号挿入位置にいずれに同期符号を挿入するために、該出力符号列中にスタッフィングビットの挿入を行うか、または前記出力符号列中に予め周期的に複数の同期符号挿入位置を定めると共に、前記多重化符号列の境界を指し示す情報（ポインタ情報）を挿入し、前記複数の同期符号挿入位置のいずれかに同期符号を挿入する。このようにすることにより、同期符号を周期的に定められた複数の同期符号挿入位置のいずれかにのみ挿入することができる。

【 0 0 3 8 】

ここで、スタッフィングビットは出力符号列の逆方向から一意に復号可能な符号であることが望ましい。このようにすると、復号化装置においてスタッフィングビットの直前の符号列の復号終了位置とスタッフィングビットの開始位置とを比較することにより、入力符号列中の誤りを容易に検出することができる。

【 0 0 3 9 】

さらに、このスタッフィングビットは同期符号およびその一部との間のハミング距離が予め定められた値以上であることが好ましい。このようにすると、擬似同期の発生確率が低くなるという利点がある。

【 0 0 4 5 】

なお、本発明において同期符号とは符号列中に同期回復のために挿入される一意に復号可能な符号であり、例えば同期符号を挿入する符号列が、フレーム単位で入力される画像信

10

20

30

40

50

号を圧縮符号化した複数種類の圧縮符号を多重化した多重化符号列であれば、符号化フレームの区切りや、複数種類の圧縮符号の区切りその他の区切りを示す符号である。

【0046】

【発明の実施の形態】

以下、図面を参照して本発明の実施形態を説明する。

【0047】

(第1の実施形態)

図1は、動き補償適応予測と直交変換符号化の一種である離散コサイン変換符号化を用いた高能率圧縮符号化装置に、本発明による誤り訂正/検出符号切り替え機能を有する符号化装置を組み合わせた動画像符号化装置の一実施形態を示すブロック図である。動き補償適応予測と離散コサイン変換符号化を組み合わせた符号化方式については、例えば文献1：安田浩編著、“マルチメディア符号化の国際標準”、丸善、(平成3年6月)等に詳しいので、動作の概略のみを説明する。また、本実施形態で用いられる誤り訂正/検出符号は、BCH符号のように情報ビットと検査ビットが分離されているものとする。

【0048】

図1において、フレーム単位で入力される符号化対象の入力動画像信号131は、まずマクロブロック等の小領域単位で動き補償適応予測が行われる。すなわち、動き補償適応予測器101において、入力動画像信号131とフレームメモリ102中に蓄えられている既に符号化/局部復号化が行われた画像信号との間の動きベクトルが検出され、この動きベクトルに基づいて動き補償予測により予測信号132が作成される。この動き補償予測器101においては、動き補償予測符号化と、入力動画像信号131をそのまま符号化するフレーム内符号化(予測信号=0)のうち、符号化に用いて好適な方の予測モードが選択され、対応する予測信号132が出力される。

【0049】

予測信号132は減算器103に入力され、入力動画像信号131から予測信号132が減算されることによって予測残差信号133が出力される。予測残差信号133は、離散コサイン変換器104において一定の大きさのブロック単位で離散コサイン変換(DCT)され、DCT係数が生成される。このDCT係数は量子化器105で量子化される。量子化器105からの量子化されたDCT係数データは2分岐され、一方は第1の可変長符号化器106で可変長符号化され、他方は逆量子化器107で逆量子化された後、逆離散コサイン変換器108で逆離散コサイン変換(逆DCT)される。逆離散コサイン変換器108からの出力は加算器109において予測信号132と加算され、局部復号信号が生成される。この局部復号信号は、フレームメモリ102に記憶される。

【0050】

一方、動き補償適応予測器101において決定された予測モードおよび動きベクトルの情報は、第2の可変長符号化器110で可変長符号化される。第1、第2の可変長符号化器106、110から出力される可変長符号(圧縮符号)は、多重化器111において多重化され、多重化符号列201となる。

【0051】

多重化器111からは、多重化符号列201と、これに対応する誤り訂正/検出符号の種類を示すFEC種類識別信号202および同期符号の挿入を要求する同期符号挿入要求信号203が出力される。

【0052】

これらの符号列202、FEC種類識別信号202および同期符号挿入要求信号203は、符号列202を誤り訂正/検出能力の異なる複数種類の誤り訂正/検出符号に切り替えて符号化する出力符号化装置200に入力され、最終的な出力符号列205が生成される。本実施形態においては、出力符号化装置200が本発明に係る符号化装置に相当する。

【0053】

図2は、多重化器111における多重化の流れを示す図である。多重化は符号化フレームを単位として行われる。まず、同期符号301が多重化される。同期符号301が多重化

10

20

30

40

50

されたときには、多重化器 1 1 1 から同期符号挿入要求信号 2 0 3 が出力され、多重化された符号語が同期符号であることが符号化装置 2 0 0 に知らされる。次に、多重化符号列 2 0 1 にその符号化フレームの各種符号化モードを示すピクチャヘッダ 3 0 2 が多重化される。次に、各領域における動き補償適応予測器 M C における予測モードを示す予測モード情報 3 0 3 が多重化され、さらに動きベクトル情報 3 0 4 および予測残差信号の D C T 係数（以下、これを残差 D C T 係数という）3 0 5 が多重化される。ピクチャヘッダ 3 0 2、予測モード情報 3 0 3、動きベクトル情報 3 0 4 および残差 D C T 係数 3 0 5 を多重化するときには、それぞれに対応する誤り訂正 / 検出符号の種類を示す F E C 種類識別信号 2 0 2 が出力される。

【 0 0 5 4 】

10

誤りが混入すると画質が大きく劣化してしまうピクチャヘッダ 3 0 2、予測モード情報 3 0 3 および動きベクトル情報 3 0 4 には、訂正 / 検出能力の高い誤り訂正 / 検出符号を用いる。一方、残差 D C T 係数 3 0 5 は、誤りが混入してもそれを検出して残差を 0 とすることで大きな画質劣化を防ぐことができるため、誤り訂正能力はそれほど高くする必要はなく、単に誤り検出を行うことができるようにするだけでもよい。

【 0 0 5 5 】

図 3 は、図 1 中の出力符号化装置 2 0 0 の構成を示すブロック図である。この出力符号化装置 2 0 0 は、ビット挿入器 2 1 1、誤り訂正 / 検出切替符号化器 2 1 2 および符号列組立器 2 1 3 から構成される。また、図 4 は出力符号化装置 2 0 0 で生成される出力符号列 2 0 5 の一例を示す図である。図 4 において、P S C は同期符号、P H はピクチャヘッダ、M O D E は予測モード情報、M V は動きベクトル情報、C H K は誤り訂正 / 検出符号の検査ビット、C O E F は残差 D C T 係数、S T U F F はスタッフィングビット（挿入ビット）である。この出力符号列 2 0 5 は、以下のような特徴を持っている。

20

【 0 0 5 6 】

（ 1 ）同期符号 P S C は、一定間隔（s y n c \_ p e r i o d ビットおき）に配置された矢印で示される同期符号挿入位置のいずれかにのみ挿入される。s y n c \_ p e r i o d の長さは、同期符号 P S C の長さで検査ビット C H K の最大長よりも大きくとる。検査ビット C H K は、同期符号挿入位置の直前に来るように位置ずらしが行われている。

【 0 0 5 7 】

（ 2 ）1 フレーム、つまり同期符号 P S C と次の P S C とで挟まれた 1 同期期間の最後の部分の誤り訂正 / 検出符号は、最後に残った情報ビットだけを符号化する縮退符号とし、検査ビット C H K（図 4 の例では C H K 6）の位置ずらしを行うために必要なビット数だけスタッフィングビット S T U F F のビット挿入を行う。

30

【 0 0 5 8 】

（ 3 ）誤り訂正 / 検出符号の種類や数を示す F E C 種類識別信号は、図 4 の出力符号列 2 0 5 中には存在しない。

【 0 0 5 9 】

この出力符号列 2 0 5 では、上記（ 1 ）のように検査ビット C H K の位置ずらしを行っているため、矢印で示される同期符号挿入位置には検査ビット C H K が入ることが無く、従って検査ビット C H K によって擬似同期が発生する可能性は全く無い。また、上記（ 2 ）のようにフレームの最後の誤り訂正 / 検出符号化を行う場合、従来の技術では多くの挿入ビット（ダミービット）を挿入することが必要であったが、本実施形態ではフレームの最後が縮退符号となっているため、挿入ビット数は少なくすむ。また、上記（ 3 ）のように誤り訂正 / 検出符号の種類や数を示すヘッダ情報を出力符号列 2 0 5 中に含ませていないため、これによる符号量の増加は無い。

40

【 0 0 6 0 】

以下、このような出力符号列 2 0 5 を生成する図 3 の出力符号化装置 2 0 0 の構成と動作について、多重化器 1 1 1 から出力される図 2 の多重化符号列 2 0 1 および図 4 の出力符号列 2 0 5 と対応させて詳細に説明する。

【 0 0 6 1 】

50

多重化器 1 1 1 で同期符号 3 0 1 が多重化されたとき、前述のように同期符号挿入要求信号 2 0 3 が出力される。同期符号 3 0 1 は、例えば図 5 のように `sync__0__len` ビットの “ 0 ”、1 ビットの “ 1 ” および同期符号 3 0 1 の種類を示す `sync__nb__len` ビットの “ x x x x x ” からなる。出力符号化装置 2 0 0 は、多重化器 1 1 1 からの同期符号 3 0 1 および同期符号挿入要求信号 2 0 3 を受けると、符号列組立器 2 1 3 から出力符号列 2 0 5 として同期符号 ( P S C ) を出力する。

#### 【 0 0 6 2 】

ここで、同期符号 3 0 1 は図 4 に示したように出力符号列 2 0 5 中の `sync__period` ビットおきに配置された同期符号挿入位置のみにしか挿入することができないようになっているため、それまでに生成された出力符号列 2 0 5 の最後尾が同期符号挿入位置に 10  
無い場合には、後述のようにスタッフィングビット `STUFF` を挿入して、同期符号 3 0 1 が同期符号挿入位置に来るようにする。

#### 【 0 0 6 3 】

同期符号 3 0 1 が出力符号列 2 0 5 に出力されると、以降ピクチャヘッダ 3 0 2、予測モード情報 3 0 3、動きベクトル情報 3 0 4、残差 D C T 係数 3 0 5 が以下のように符号化される。まず、多重化器 1 1 1 から出力された多重化符号列 2 0 1 に対して、ビット挿入器 2 1 1 において擬似同期の発生を防ぐためのビット挿入が行われる。すなわち、出力符号列 2 0 5 中に同期符号 3 0 1 の符号語と同一のビットパターンがあると同期符号 3 0 1 を一意に復号できなくなることを防ぐために、必要に応じてビット挿入を行う。例えば、同期符号 3 0 1 が図 5 のように `sync__0__len` ビットの “ 0 ” が連続する符号語 20  
ならば、同期符号 3 0 1 以外の符号列中に `sync__0__len` ビット以上 “ 0 ” が連続しないように “ 1 ” を挿入すれば、擬似同期の発生を防ぐことができる。

#### 【 0 0 6 4 】

前述のように同期符号 3 0 1 は同期符号挿入位置にしか挿入されないため、この擬似同期の発生防止のためのビット挿入操作は同期符号挿入位置だけで行えばよい。そこで、これまでに生成された出力符号列 2 0 5 の総ビット数を示す計数値 2 2 1 を符号列組立器 2 1 3 から出力し、ビット挿入器 2 1 1 で計数値 2 2 1 に基づいてビット挿入が必要であるか 30  
否かを判定する。計数値 2 2 1、つまり既に生成された出力符号列 2 0 5 の総ビット数を `total__len` とすれば、

$0 < \text{total\_len} \bmod \text{sync\_period} \text{ sync\_0\_len}$  30  
となる区間で多重化符号列 2 0 1 中の “ 1 ” の数を計数し、この区間に “ 1 ” が全くなければ 1 ビットの “ 1 ” を挿入する。ここで、 $A \bmod B$  は A を B で除したときの剰余を表す。

#### 【 0 0 6 5 】

また、誤りによる同期符号 3 0 1 の誤検出の確率を低くするには、以下のようにビット挿入を行えばよい。

#### 【 0 0 6 6 】

同期符号 3 0 1 に n ビット誤りが混入した場合にも、同期符号 3 0 1 を検出するためには、後述する動画像復号化装置の入力復号化装置で真の同期符号とハミング距離が n 以下の符号語は同期符号と判定する必要がある。しかし、同期符号 3 0 1 以外の符号列をそのままにしてこのような判定を行うと、同期符号 3 0 1 以外の符号列中にも同期符号とハミング距離が n 以下のビットパターンが存在することがあり、これが同期符号挿入位置にあると誤まって同期符号 3 0 1 と判定されてしまう。 40

#### 【 0 0 6 7 】

そこで、ビット挿入器 2 1 1 で多重化符号列 2 0 1 に以下のようにビット挿入を行うことによって、多重化符号列 2 0 1 中の同期符号挿入位置にある同期符号以外の符号列を同期符号 3 0 1 とのハミング距離が  $2 * n + 1$  以上離れた値となるように変換する。具体的には、

$0 < \text{total\_len} \bmod \text{sync\_period} \text{ sync\_0\_len}$   
- (  $2 * N + 1$  ) 50

となる区間での“1”の数(=  $n_0$  とする)を計数し、 $n_0$  が  $2 * n + 1$  以下であったら  $2 * n + 1 - n_0$  ビットの“1”を多重化符号列 201 に挿入する。

【0068】

こうしてビット挿入器 211 でビット挿入が行われた後の符号化列 222 は、誤り訂正 / 検出符号の種類を示す FEC 種類識別信号 202 と共に、誤り訂正 / 検出符号切替符号化部 212 に入力される。

【0069】

図6は、図3中の誤り訂正 / 検出符号切替符号化部 212 の構成を示すブロック図である。ラッチ回路 603 は、FEC 種類識別信号 202 をラッチする回路であり、多重化器 111 から多重化符号列 201 への同期符号の出力が終了して同期符号挿入要求信号 203 が停止すると、FEC 種類識別信号 202 をラッチし、ラッチした信号 623 を誤り訂正 / 検出符号化器 604 に供給する。

【0070】

誤り訂正 / 検出符号化器 604 は、ラッチされた信号 623 に従ってビット挿入器 211 からの符号列 222 を誤り訂正 / 検出符号化し、情報ビット 631 および検査ビット 632 を生成して出力する。また、誤り訂正 / 検出符号化器 604 は1ブロックの誤り訂正 / 検出符号化が終了すると、ラッチ回路 603 に次の FEC 種類識別信号 202 のラッチを指示するラッチ指示信号 625 を出す。ラッチ回路 603 は、このラッチ指示信号 625 に従ってラッチを行い、ラッチした信号 623 を誤り訂正 / 検出符号化器 604 に供給する。

【0071】

出力符号化装置 200 では以上の動作が繰り返され、ビット挿入器 211 からのビット挿入後の符号列 222 は、誤り訂正 / 検出切替符号化器 212 で多重化器 111 から示された FEC 種類識別信号 202 に従って誤り訂正 / 検出符号の切り替えがなされながら、誤り訂正 / 検出符号化が行われる。FEC 種類識別信号 202 は、1ブロックの誤り訂正 / 検出符号の符号化の終了時点でのみラッチ回路 603 でラッチされるため、この切り替え点までは同一の誤り訂正 / 検出符号が適用される。例えば、ピクチャヘッダ 302 は FEC 1、予測モード情報 303 は FEC 2 という誤り訂正 / 検出符号を用いる場合、ピクチャヘッダ 302 のビット数が FEC 1 の1ブロックの情報ビット数より短い場合は、後続の予測モード情報 303 の誤り訂正 / 検出符号としては、FEC 1 の情報ビット数に達するまでは FEC 1 が用いられる。

【0072】

図7は、図3中の符号列組立器 213 の構成を示すブロック図である。この符号列組立器 213 は、出力符号列 205 のビット数を計数するカウンタ 701、検査ビット 632 とそのビット数を一時的に記憶するバッファ 702、出力符号列 205 を切り替える切替器 703、および切替器 703 を制御する切替制御器 704 から構成される。

【0073】

カウンタ 701 は、同期符号挿入要求信号 203 が入力すると同期符号長 `sync_len` の値にリセットされ、同期符号の次のビットから順に次の同期符号が入力するまでカウントアップしていく。同期符号入力後、最初の検査ビット 632 が入力するまでは情報ビット 631 が出力されるように切替器 703 が動作する。検査ビット 632 が入力すると、バッファ 702 に記憶されると共に、そのビット数(検査ビット数) 711 がバッファ 702 から切替制御器 704 に出力される。

【0074】

切替制御器 704 は、検査ビット数 711 とカウンタ 701 の計数値 221 を基に、前述のように検査ビット 632 が同期符号挿入位置に出力されないようにするための検査ビットずらしを行うように切替器 703 を制御する。例えば計数値 221 を `bit_count`、検査ビット数 711 を `check_len` とすれば、  
`bit_count mod sync_period < sync_period - check_len`

10

20

30

40

50



の時は情報ビット631を出力し、

$$\text{sync\_period} - \text{check\_len} \quad \text{total\_bits} \bmod \text{sync\_period} < \text{sync\_period}$$

の時には、バッファ702に記憶された検査ビット713を出力する。以降、情報ビット631と検査ビット632を入力しながら、以上の処理を繰り返す。

#### 【0075】

出力符号化装置200は、前述のように1フレームの最後の部分では誤り訂正/検出符号として縮退符号を用いることと、検査ビットの位置ずらしを行うためにビット挿入を行うため、それ以外の部分の通常とは異なった動作を行う。すなわち、多重化器111は1フレームの多重化符号列201の出力が終了すると、次のフレームの同期符号挿入要求信号203を出力する。これに対応して、図6の誤り訂正/検出切替符号化部212内の誤り訂正/検出符号化器604は、誤り訂正/検出符号の情報ビット631の不足分は挿入ビット発生器705から出力される予め定められたビットパターンであるとみなして、縮退符号を用いた誤り訂正/検出符号化を行う。このビットパターンは、全ビットを“1”としてもよいし、“0”としてもよいし、“0101……”のような特定のパターンの繰り返しでもよい。情報ビット631には、この補われた挿入ビットは出力しない。

#### 【0076】

図7の符号列組立器213では、情報ビット631を最後まで出力した後、切替器703をビット発生器705から入力に切り替え、バッファ702に記憶されている検査ビット713が次の同期符号の直前に配置されるように挿入ビットを挿入する。この挿入ビットの数  $\text{stuffing\_len}$  は、1フレームの最後の情報ビット631を出力したときのカウンタ701の計数値221を  $\text{total\_len}$  とし、最後に出力する検査ビット632のビット数を  $\text{last\_check\_len}$  とすれば、

$$\text{stuffing\_len} = \text{sync\_period} - \text{last\_check\_len} - (\text{total\_len} \bmod \text{sync\_period})$$

となる。なお、縮退符号を用いない場合は、最後の情報ビット  $\text{last\_into\_len}$  の通常の情報ビット  $\text{into\_len}$  からの不足分  $(\text{into\_len} - \text{last\_into\_len})$  ビットのビット挿入を行い、さらに検査ビットの位置ずらしを行うためのビット挿入を行う必要がある。このため、縮退符号を用いる場合に比べ  $\text{into\_len} - \text{last\_into\_len} + (\text{into\_len} - \text{last\_into\_len}) \bmod \text{sync\_period}$  ビット多くの挿入ビットが必要となってしまう。

#### 【0077】

符号列組立器213は、こうして切替器703を介して情報ビット631および挿入ビットを出力符号列205に出力した後、最後に検査ビット713に切り替えて出力符号列205に出力する。

#### 【0078】

次に、本実施形態に係る動画像復号化装置について説明する。

#### 【0079】

図8は、図1の動画像符号化装置に対応した動画像復号化装置の構成を示すブロック図である。図1の動画像符号化装置から出力された出力符号列205は伝送/蓄積系を経た後、入力符号列205として入力復号化装置800に入力される。本実施形態においては、入力復号化装置800が本発明に係る復号化装置に相当する。

#### 【0080】

入力復号化装置800では、後続の逆多重化器811からの誤り訂正/検出符号の種類を示すFEC種類識別信号802に基づいて誤り訂正/検出符号を切り替えながら、誤り訂正/検出復号化された符号列801、同期符号検出信号803および誤り検出信号804が出力される。逆多重化器811は、これらの符号列801、同期符号検出信号803および誤り検出信号804を入力し、予測残差符号841と動き補償適応予測情報符号842を分離して出力する。

10

20

30

40

50

## 【 0 0 8 1 】

予測残差符号 8 4 1 および動き補償適応予測情報符号 8 4 2 は、それぞれ第 1 および第 2 の可変長復号器 8 0 6 および 8 1 0 に入力される。第 1 の可変長復号器 8 0 6 で復号された残差 D C T 係数 8 3 1 は、逆量子化器 8 0 7 で逆量子化、逆 D C T 器 8 0 8 で逆離散コサイン変換という一連の処理が施された後、加算器 8 0 9 で動き補償適応予測器 8 0 1 からの出力である動き補償適応予測信号 8 3 2 と加算されて再生画像信号 8 5 0 となる。再生画像信号 8 5 0 は、装置外へ出力されると共に、フレームメモリ 8 2 0 へ記録される。第 2 の可変長復号器 8 1 0 で復号された動き補償適応予測情報は、動き補償適応予測器 8 0 1 に入力され、動き補償予測信号 8 3 2 が生成される。

## 【 0 0 8 2 】

10

以上の処理は、図 1 の動画像符号化装置に対応して動画像を再生する処理であり、逆量子化器 8 0 7、逆 D C T 器 8 0 8、加算器 8 0 9 およびフレームメモリ 8 2 0 が行う処理は、それぞれ図 1 における逆量子化器 1 0 7、逆 D C T 器 1 0 8、加算器 1 0 9 およびフレームメモリ 1 0 2 のそれと実現手段は異なる場合もあるが、基本的に同一である。また、第 1 および第 2 の可変長復号器 8 0 6、8 1 0、逆多重化器 8 1 1 および入力復号化装置 8 0 0 の処理は、符号列に誤りが混入した場合を除いて、それぞれ図 1 における可変長符号化器 1 0 6、1 1 0、多重化器 1 1 1 および出力符号化装置 2 0 0 のそれと逆の処理である。

## 【 0 0 8 3 】

図 9 は、入力復号化装置 8 0 0 の構成を示すブロック図である。この入力復号化装置 8 0 0 は、入力符号列 2 0 5 中の同期符号を検出する同期検出器 9 0 1、入力符号列 2 0 5 のビット数を計数するカウンタ 9 0 2、入力符号列 2 0 5 を情報ビット 9 1 2 と検査ビット 9 1 3 に分解して出力する符号列分解器 9 0 3、誤り訂正 / 検出復号化器 9 0 4 および挿入ビット除去器 9 0 5 から構成される。

20

## 【 0 0 8 4 】

同期検出器 9 0 1 では、カウンタ 9 0 2 の計数値 9 1 1 に基づいて同期符号挿入位置でのみ同期符号の検出を行う。例えば、同期符号挿入位置の間隔を `sync__period`、計数値 9 1 1 を `bit__count` とし、同期符号の長さを `sync__len` とすれば、

`0 < bit__count % sync__period sync__len`

30

の時のみ同期検出を行う。

## 【 0 0 8 5 】

ここで、同期符号の誤りを考慮した同期符号の検出を行っても良い。

## 【 0 0 8 6 】

図 3 中の出力符号化装置 2 0 0 内のビット挿入器 2 1 1 において、 $n$  ビット以下の誤りを考慮して同期符号とのハミング距離が  $2 * n + 1$  離れるようにビット挿入による符号列変換を行っていれば、真の同期符号とのハミング距離が  $n$  以下のものを同期符号と判定しても、 $n$  ビット以下の誤りの混入であれば誤同期検出は起こらない。

## 【 0 0 8 7 】

図 1 0 は、符号列分解器 9 0 3 の構成を示すブロック図である。入力符号列 2 0 5 は、後述する制御器 1 0 0 1 で制御される第 1 の切替器 1 0 0 2 で情報ビット 1 0 2 1 と検査ビット 9 1 3 とに切り替えられる。第 1 の切替器 1 0 0 2 から情報ビット 1 0 2 1 が出力された場合、この情報ビット 1 0 2 1 は第 2 の切替器 1 0 0 3 を経てバッファ 1 0 0 4 に情報ビット長だけ記憶される。カウンタ 1 0 0 5 は、第 2 の切替器 1 0 0 3 からの出力ビット数を計数する。このカウンタ 1 0 0 5 の計数値 1 0 2 3 は、比較器 1 0 0 6 で誤り訂正 / 検出符号情報出力器 1 0 0 7 から出力された情報ビット長 1 0 2 4 と比較され、両者が一致したときにカウンタ 1 0 0 5 がリセットされると共に、誤り訂正 / 検出符号の種類を示す F E C 種類識別信号 8 0 2 がラッチ回路 1 0 0 8 でラッチされ、さらにバッファ 1 0 0 4 から情報ビット 9 1 2 が出力される。ラッチ回路 1 0 0 8 の出力 9 1 4 は、誤り訂正 / 検出符号情報出力回路 1 0 0 7 へ入力されると共に、図 9 中に示す誤り訂正 / 検出復

40

50

号化器 904 に出力される。

#### 【0088】

前述のように、誤り訂正／検出符号の検査ビットは位置ずらしが施され、符号列 205 中後方の誤り訂正／検出符号の情報ビットの間に入っている。制御器 1001 は、この位置ずらしが施された検査ビットと情報ビットが分離されるように制御を行う。1 ブロックの誤り訂正／検出符号の情報ビットの入力が終了すると、比較器 1006 において計数値 1023 と情報ビット長 1024 が一致する。制御器 1001 は、この一致信号を受けて誤り訂正／検出情報出力器 1007 から検査ビット長 1025 を取り込み、次の情報ビットの間に入っている検査ビットの位置を計算する。比較器 1006 で一致と判定されたときの符号列 205 の入力ビット数の計数値 911 を `bit __count`、検査ビット長を `check __len` とすれば、検査ビット開始位置 `check __start` は

$$\text{check\_start} = (\text{bit\_count} / \text{sync\_period} + 1) * \text{sync\_period} - \text{check\_len}$$

となり、検査ビット終了位置 `check __end` は、

$$\text{check\_end} = (\text{bit\_count} / \text{sync\_period} + 1) * \text{sync\_period}$$

となる。制御器 1001 は、計数値 911 が `check__start` から `check__end` までの間、切替器 1002 を検査ビット 913 が出力されるように制御する。

#### 【0089】

1 フレームの最後は縮退符号で誤り訂正／検出符号化が行われているので、特別な処理を行う。1 フレームの最後に達すると、同期検出器 901 から次のフレームの同期符号を検出したことを示す信号 803 が出力される。制御器 1001 は、この信号 803 を受けてフレームの最後の誤り訂正／検出符号の検査ビットの位置および情報ビットの不足ビット数を計算する。1 フレームの最後の誤り訂正／検出符号を入力し始めたときの入力される符号列 205 のビット数の計数値 911 を `pre__last__count`、1 フレームの符号列 205 の入力が終了した時点の計数値 911 を `total__count`、処理時点での計数値 911 を `bit__count`、1 フレーム最後の誤り訂正／検出符号の検査ビット長を `last__check __len`、それより一つ前の誤り訂正／検出符号の検査ビット長を `pre __last__check __len` とする。まず、誤り訂正符号が縮退符号であることおよびビット挿入が行われていることによる情報ビットの過不足分を計算する。1 フレーム最後の誤り訂正／検出符号の情報ビットのうち、出力符号列 205 中に含まれているビット数 `last__info__len` は、

$$\text{last\_info\_len} = \text{total\_count} - \text{last\_check\_len} - \text{pre\_last\_count} - \text{pre\_last\_check\_len}$$

である。`last__info__len` が誤り訂正符号の情報長 `info__len` より短いときには縮退符号であると判定し、計数値 1023 が `last__info__len` から `info__len` までは切替器 1021 を挿入ビット発生器 1015 から出力されるビットパターンに切り替えて、縮退による情報ビットの不足分を補う。この挿入ビット発生器 1015 からの出力ビットパターンは、符号化器の図 7 中の挿入ビット発生器 705 と同一のビットパターンを発生する。

#### 【0090】

一方、`last__info__len` が `info__len` より長いときには、ビット挿入分と判定し計数値 1023 が `info__len` 以上となる部分については情報ビット 912 を出力しないようにする。検査ビットについては、

$$\text{total\_count} - \text{check\_len} < \text{bit\_count} \quad \text{total\_count}$$

の時の出力符号列 205 を検査ビットとして出力するように切替器 1002 を制御する。

## 【0091】

誤り訂正／検出復号化器904は、符号列分解器903から出力された情報ビット912および検査ビット913を入力し、図10のラッチ回路1008でラッチされた誤り訂正／検出符号の種類を示すFEC種類識別信号914に基づいて誤り訂正／検出符号の復号化を行い、誤り訂正された符号列915と誤り検出信号804を出力する。

## 【0092】

誤り訂正された符号列915は、挿入ビット除去器905に入力される。挿入ビット除去器905は、出力符号化装置200のビット挿入器211で挿入された擬似同期信号を防ぐための挿入ビットを取り除く処理を行う。前述のように、ビット挿入は同期挿入位置でのみ行われているため、計数器902の計数値911に基づいて同期挿入位置を判定する

10

## 【0093】

例えば、同期符号語が図5に示すものとし、ビット挿入器211において同期符号とのハミング距離が $2 * n + 1$ 以上離れるように同期符号の最初の`sync__len`ビットの“0000...”の部分にビット挿入が行われている場合には、同期符号挿入位置から`sync__len - (2 * n + 1)`ビットの“1”の数(=  $n_0$ とする)を計数し、 $n_0$ が $2 * n + 1$ 以下であったら $2 * n + 1 - n_0$ ビット削除する。ただし、挿入ビットは“1”であると定められているため、挿入ビット除去器905で挿入ビットと判定されたビットが“0”となっていたときは、同期符号挿入区間に誤りが混入したと考えられ、この場合には誤り検出信号804を出力する。

20

## 【0094】

以上のようにして入力復号化装置800で復号された符号列801は、逆多重化器811で逆多重化が行われる。これは図2に示したような多重化された符号語を分離して出力する操作である。この逆多重化器811は、第1および第2の可変長復号器806および810と連動して動作する。

## 【0095】

まず、出力復号化装置800から同期符号検出信号803が入力されると、逆多重化器811はフレーム処理の初期状態に移る。次に、誤り訂正／検出符号の種類を示すFEC種類識別信号802としてピクチャヘッダに対する誤り訂正／検出符号の種類を出力し、符号列801を入力してピクチャヘッダ302を復号しピクチャヘッダに誤りが無いか判定

30

## 【0096】

第2の可変長復号化器810は、全ての予測モード情報を復号すると、その旨を示す信号を逆多重化器811に出力する。これを受けて、逆多重化器811は動きベクトル情報304に対する誤り訂正／検出符号の種類を示すFEC種類識別信号を出力し、動きベクトル情報304の逆多重化を開始する。逆多重化された動きベクトル情報は第2の可変長復号器810に出力され、復号化される。全ての動きベクトル情報の復号が終了すると、第2の可変長復号器810から逆多重化器811にその旨を示す信号が出力され、これを受けて逆多重化器811は残差DCT係数305に対する誤り訂正／検出符号の種類を示すFEC種類識別信号を出力すると共に、残差DCT係数を305を逆多重化して第1の可変長復号器806に出力する。この残差DCT係数305を第1の可変長復号器806で復号する。

40

## 【0097】

以上のように、誤り訂正／検出符号の種類は、逆多重化器811において出力符号化装置200と同一に定められた多重化の規則に基づいて判定される。このため、誤り訂正／検出符号の種類を示すヘッダ情報等を出力符号列205に含める必要はない。

## 【0098】

誤り訂正／検出復号化器904において、誤り検出符号により入力符号列205に誤り

50

が混入していることが検出されることがある。また、前述のように挿入ビット除去器 9 0 5 で挿入ビットの誤りが検出されることもある。これらの場合には、入力復号化装置 8 0 0 から誤り検出符号 8 0 4 が出力される。さらには、可変長復号化の処理において可変長符号語テーブルに無い符号語が検出された場合にも、誤りが混入したと判定される。また、逆多重化器 8 1 1 における逆多重化処理において多重化の規則に違反する部分があると判定されたときも、誤りが混入したと判定される。これらの場合、入力復号化装置 8 0 0 および逆多重化器 8 1 1 は再生画像に大きな劣化が生じないように、以下のような処理を行う。

#### 【0099】

(1) 残差 DCT 係数で誤りが検出された場合は、当該部分の残差を 0 とする。予測モードとしてイントラ符号化モードが選択されていた場合は、既再生フレームや周囲の領域の再生画像信号から当該領域の再生画像信号を予測してもよい。

10

#### 【0100】

(2) 予測モード情報や動きベクトルで誤りが検出された場合は、周囲の領域の予測モード情報や動きベクトル情報から当該領域の予測モード情報や動きベクトル情報が推定できるときにはそれらを用い、それが不可能ならば既再生フレームや周囲の領域の再生画像信号から当該領域の再生画像信号を予測する。

#### 【0101】

(3) ピクチャヘッダに誤りが検出された場合は、それをそのまま復号すると非常に大きな画質劣化を生じる可能性があるため、前フレームの再生画像をそのまま現フレームの再生画像とする。

20

#### 【0102】

以上の(1)(2)(3)処理において、可変長符号化を用いているために誤りが次の同期符号までの後続の符号にも波及している場合には、その部分についても同様の処理を行う。

#### 【0103】

上述の説明では、同期符号検出器 9 0 1 が同期符号挿入位置 (sync\_\_period ビットおき) でのみ同期符号の検出を行う例を示したが、伝送 / 蓄積媒体によってはビットの消失や誤ビットの挿入が起こる場合もある。このような場合は、同期符号挿入位置以外でも同期符号の検出を行い、同期符号が検出された位置を同期符号挿入位置と判定すればよい。

30

#### 【0104】

(第2の実施形態)

次に、図 1 2 ~ 図 1 4 を参照して本発明に係る第 2 の実施形態について説明する。本実施形態による動画像符号化装置および動画像復号化装置は、ビット列の一部が消失してビット数が減少したり、余分なビットが付加されてビット数が増加するような伝送路 / 蓄積媒体で符号列の伝送 / 蓄積を行っても、同期検出を確実に行うことができるようにしたものである。

#### 【0105】

図 1 2 は、このようなビット付加 / 消失のある場合の同期検出の処理の原理を示すブロック図である。ここで、正しい同期符号は図 1 2 (a) に示すように、sync\_\_0\_\_1 len ビットの“0”と、1 ビットの“1”からなるとする。なお、図 1 2 中の“x”は同期符号以外のビットである。

40

#### 【0106】

図 1 2 (b) ~ (e) は、ビット付加 / 消失によって同期符号がどのように変化するかを示したものである。ここで、付加 / 消失するビット数 (Nid) は最大 1 ビットとしている。(b) は同期符号より前のビット列中で 1 ビット削除が起きた場合であり、同期符号全体が 1 ビット前にシフトしている。(c) は同期符号より前のビット列中で 1 ビット付加が起きた場合であり、同期符号全体が 1 ビット後ろにシフトしている。(d) は同期符号中でビット削除が起きた場合であり、図中矢印のビット抜け位置から後ろが 1 ビット前

50

にシフトしている。さらに、(e)は同期符号中で1ビット付加が起きた場合であり、図中矢印のビット付加位置に1ビット付加が起こり、それより後ろが1ビット後ろにシフトしている。

#### 【0107】

ビット付加/消失が起きても正しく同期を検出するためには、図12(b)~(d)に示したビット列も同期符号と判定する必要がある。図12から分かるように、正しい同期符号挿入位置の $\pm Nid$ ビットの範囲に含まれる“1”の数は最大

$$sync\_0\_len - 3 * Nid$$

ビットである。従って、復号側で同期符号挿入位置の $\pm Nid$ ビットの範囲で同期検出を行い、この区間に含まれる“1”の数が上記の値以下だったら同期符号と判定すればよい。また、符号化装置では図12(b)~(d)のビットパターンが生じないように符号列の変換を行う。

10

#### 【0108】

以下、このような符号化装置/復号化装置を第1の実施形態との差異を中心に説明する。

#### 【0109】

第2の実施形態に係る動画像符号化装置は、第1の実施形態に係る動画像符号化装置と全体構成は同じであるが、図3中のビット挿入器211の動作が異なっている。図13に、ビット挿入器211の動作を示す。すなわち、第1の実施形態のビット挿入器211では、同期符号挿入区間でのみビット挿入操作を行っていたが、第2の実施形態におけるビット挿入器211では、最大 $Nid$ ビットのビット付加/消失が起きても同期符号と同一のビットパターンが生じないようにするため、同期符号挿入区間 $\pm Nid$ ビットの区間でビット挿入を行う。

20

#### 【0110】

図3における計数値221を  $total\_len$ 、同期符号挿入位置の間隔を  $sync\_period$  とすれば、ビット挿入器211は

$$total\_len \bmod sync\_period = sync\_period - Nid$$

(mod: 剰余演算)

から、

$$total\_len \bmod sync\_period = sync\_0\_len - 1 - 3 * Nid \quad 30$$

の区間での“1”の数(=  $n0$  とする)を計数し、 $n0$ が  $3 * Nid + 1$  未満であったら  $3 * Nid + 1 - n0$  ビットの“1”を挿入する。

#### 【0111】

図13では、 $sync\_period = 12$ 、 $sync\_0\_len = 9$ 、 $Nid = 1$  の場合のビット挿入器211の動作例を示している。この例では  $n0 = 2$  であるので、 $3 * Nid + 1 - n0 = 2$  ビットの“1”を挿入する。

#### 【0112】

このようにビット挿入を行うことにより、同期符号挿入区間 $\pm Nid$ ビットの“0”の数は  $3 * Nid$  ビット以上であることが保証され、同期符号と一意に識別することが可能である。

40

#### 【0113】

一方、第2の実施形態に係る動画像復号化装置は、第1の実施形態と全体構成は同一であるが、図9中の同期検出器901および挿入ビット除去器905の動作が異なっている。図14に、挿入ビット除去器905の動作を示す。

#### 【0114】

すなわち、同期検出器901では最大 $Nid$ ビットのビット付加/消失が起きても同期を検出するために、同期符号挿入位置の前後 $\pm Nid$ ビットの範囲で同期符号の検出を行う。

#### 【0115】

50

まず、同期符号挿入位置毎に同期符号が存在するかを判定する。すなわち、カウンタ 902 の計数值 911 を  $bit\_count$  すると、

$bit\_count \bmod sync\_period = sync\_period - Nid$

から、

$bit\_count \bmod sync\_period = sync\_0\_len - 1 + Nid$

の間の“0”の数(=  $n0$  とする)をカウントし、 $n0$  が  $3 * Nid$  以下であたら、この区間に同期符号があると判定する。

【0116】

10

図14の動作例では、 $sync\_period = 12$ 、 $sync\_0\_len = 9$ 、 $Nid = 1$  の場合の例を示している。この例では( $bit\_count \bmod sync\_period$ )が“11”から“8”の間で“0”の数をカウントする。図14の例では  $n0 = 2$  であるので、同期符号と判定する。

【0117】

次に、同期符号があると判定された同期符号挿入区間で、ビット付加/消失によって符号列が何ビットシフトしているかを判定する。図14のような  $sync\_0\_len$  ビットの場合、最後の“1”の位置からシフト量を判定する。具体的には、同期符号判定区間の最初から  $sync\_0\_len + 1$  ビット目から見て最初にある“1”を探索し、この同期符号判定区間の最初から何ビット目にあるか(=  $first\_1\_pos$  ビットとする)を基に、

20

シフトビット数 =  $first\_1\_pos - (sync\_0\_len + 1 + Nid)$

(負の場合は前方シフト、正の場合は後方シフト)

として求める。図14の例では  $first\_1\_pos = 10$  であるから、

シフトビット数 =  $10 - (9 + 1 + 1) = -1$

となり、前方に1ビットシフトしていることが分かる。

【0118】

挿入ビット除去器 905 では、第1の実施形態と異なり、同期符号挿入位置  $\pm Nid$  ビットの区間で挿入ビット除去処理を行う。すなわち、

30

$bit\_count \bmod sync\_period = sync\_period - Nid$

から、

$bit\_count \bmod sync\_period = sync\_0\_len - 1 - 3 * Nid$

の区間での“1”の数(=  $n0$  とする)を計数し、 $n0$  が  $3 * Nid + 1$  以下であつたら、 $3 * Nid + 1 - n0$  ビットの“1”を除去する。

【0119】

第2の実施形態において、伝送路や蓄積媒体でのビット付加/消失が起きた区間を何らかの形で判定できる場合、ビット付加/消失を考慮した同期検出処理やビット挿入処理、ビット除去処理は、その区間でのみ行うようにしてもよい。

40

【0120】

なお、前述した第1の実施形態に係る動画像復号化装置においても、伝送路や蓄積媒体でのビット付加/消失に対応した同期検出を行うために、同期検出器 901 では第2の実施形態と同様に同期符号挿入区間  $\pm Nid$  ビットの区間で同期検出を行うようにしてもよい。この場合、同期符号以外の部分を誤って同期符号と判定する疑似同期が生じる可能性があるが、ビット付加/消失の起きやすい伝送路/蓄積媒体では同期検出ミスによる復号画像の品質劣化を抑えることができ、画質が向上する。

【0121】

また、伝送路や蓄積媒体でのビット付加/消失が起きた区間を何らかの形で判定できる場

50

合は、この処理はその区間でのみ行い、それ以外の区間では通常の同期検出を行うようにしてもよい。

【0122】

さらに、上述した第1および第2の実施形態において、フレームの長さを示す情報（以下、フレーム長情報という）を用いて同期をさらに保護するようにしてもよい。図15、図16および図17は、フレーム長情報P O I N T E Rを用いた場合の符号列の例を示したものである。

【0123】

図15の例では、同期符号P S Cの直後にフレーム長情報P O I N T E Rと、このフレーム長情報P O I N T E Rを保護する誤り訂正／検出符号のチェックビットC H K Pが続いている。フレーム長情報P O I N T E Rには前フレームのビット数、すなわち前フレームの同期符号から現フレームの同期符号までのビット数を示す情報が記録されている。

10

【0124】

符号化装置では、1フレームの符号列ビット数をカウントして、これをフレーム長情報P O I N T E Rに変換し、さらに誤り訂正／検出符号化を行ってチェックビットC H K Pを生成する。そして、これらを図15に示すように次フレームの同期符号の直後に続けて符号列を生成する。

【0125】

一方、復号化装置では、第1および第2の実施形態に述べたのと同様の方法で同期符号の検出を行った後、それに続くフレーム長情報P O I N T E RとチェックビットC H K Pを符号列から取り出し、誤り訂正／検出復号を行ってフレーム長情報P O I N T E Rを復号する。そして、復号されたフレーム長情報P O I N T E Rと、1つ前に検出された同期符号から現在の同期符号の位置までのビット数をカウントして求めた値（フレーム長カウント値）を比較し、同期符号の誤検出が無いかをチェックする。

20

【0126】

もし、フレーム長カウント値がフレーム長情報P O I N T E Rに示された前フレームの符号長と異なる場合には、同期符号の誤検出が起こった可能性があるため、フレーム長情報P O I N T E Rを用いて誤検出された同期符号の再検出を行う。すなわち、現在の同期符号からフレーム長情報P O I N T E Rで示されたビット数だけ前方に検出できなかった同期符号があると見なすこととする。この場合、1つ前に検出された同期符号から現在の同期符号までの区間は、1つ前の同期符号からフレーム長情報P O I N T E Rで示される位置までの区間と、そこから現在の同期符号までの区間の二つのフレームに分割して復号処理を行う。

30

【0127】

ただし、フレーム長情報P O I N T E Rで示されるビット数が、1つ前に検出された同期符号から現在の同期符号の位置までのビット数よりも多い場合には、フレーム長情報P O I N T E Rが誤っているものと見なし、上記の同記再検出処理は行わない。

【0128】

フレーム長情報P O I N T E RおよびチェックビットC H K Pのビット数が多い場合には、図16に示すように同期符号P S C、フレーム長情報P O I N T E R、チェックビットC H K Pが複数の同期区間にわたるようにしてもよい。この場合、同期符号以外の符号列が同期符号と一定のハミング距離を保つようにするための符号化装置でのビット挿入処理および復号化装置でのビット削除処理は、フレーム長情報P O I N T E RおよびチェックビットC H K Pが存在する区間では行わないようにしてもよい。

40

【0129】

図15および図16の例において、同期符号P S Cの後半にその同期符号の種類を示す情報（フレーム同期符号、G O B同期符号の区別等）が含まれている場合、フレーム長情報P O I N T E Rだけでなく、同期符号P S Cの後半ビットを誤り訂正符号で保護するようにしてもよい。これにより、同期符号の位置だけでなくその種類も正確に検出できるため、誤りに対する耐性がさらに向上する。

50



## 【0130】

図17の例では、フレーム長情報P O I N T E RおよびチェックビットC H K Pをフレームの最後(次のフレームの同期符号の直前)に置く例である。この場合、復号化装置では、次のフレームの同期符号を検出した後にその直前のフレーム長情報P O I N T E RおよびチェックビットC H K Pを取り出して誤り訂正/検出復号を行い、図15および図16と同様の処理により同期符号の再検出を行う。

## 【0131】

図15の例においては、同期符号は同期符号挿入位置にのみ存在するので、フレーム長情報P O I N T E Rはフレームのビット数を同期符号挿入間隔(= s y n c \_ p e r i o d ビット)で割った値を記録してもよい。これにより、フレーム長を少ないビット数で表すことができる。

10

## 【0132】

また、第1および第2の実施形態では、誤り訂正/検出符号を符号化する情報の重要度に応じて変化させる階層符号化を行う例を示したが、フレーム内で同一の誤り訂正/検出符号を用いるようにしてもよいし、誤り訂正/検出符号を用いなくともよい。そのような場合でも、本実施形態で示したような同期符号以外の符号列が同期符号と予め定められた値以上のハミング距離を保つためのビット挿入処理、およびこれに対応した同期符号検出処理によって、同期を検出する能力は従来方式より向上する。

## 【0133】

また、以上の各実施形態の説明では動画像信号を高能率圧縮符号化して伝送/蓄積する例を示したが、本発明は静止画像や音声、データ等の伝送/蓄積にも応用することが可能である。例えば、静止画像信号を直交変換を用いて高能率圧縮符号化する場合には、変換係数の低域成分をより強く誤り保護するように誤り訂正/検出符号を切り替えて用いればよい。音声を駆動源と声道フィルタにモデル化して符号化する方法では、ピッチ周期や声道パラメータ等を強く誤り保護するように誤り訂正/検出符号を切り替えれば良い。

20

## 【0134】

(第3の実施形態)

次に、本発明に係る第3の実施形態について説明する。本実施形態は、誤り訂正/検出符号を用いないことが第1および第2の実施形態と異なる。

## 【0135】

図18は、本実施形態に係る動画像符号化装置のブロック図である。図1と相対応する部分に同一符号を付して第1の実施形態との相違点を中心に説明すると、本実施形態では出力符号化装置200の構成および動作が異なる。また、多重化器111は、基本動作は図1の多重化器111と同一であるが、誤り訂正/検出符号を用いないことに伴って、多重化符号列201および同期符号挿入要求信号203のみを出力する。

30

## 【0136】

図19は、図18中の出力符号化装置200の構成を示すブロック図である。この出力符号化装置200は、出力符号列205のビット数を計数するカウンタ1701、出力符号列205を切り替える切替器1703、切替器1703を制御する切替制御器1704、およびスタッフィングビットを発生するスタッフィングビット発生器1705から構成される。

40

## 【0137】

図20は、図19の出力符号化装置200で生成される出力符号列205の一例を示す図である。図4の出力符号列と相対応する符号語には同一記号を用いて表記してある。図4と同様に同期符号P S Cは周期的に、つまり一定区間(s y n c \_ p e r i o d ビット)おきに配置された矢印で示される同期符号挿入位置のいずれかのみ挿入されている。図20では、誤り訂正/検出符号の検査ビットC H Kが含まれないところが図4と異なる。出力符号列205の1フレームの最後の部分には、同期符号P S Cが同期符号挿入位置に挿入されるように、スタッフィングビットS T U F Fが挿入される。スタッフィングビットS T U F Fのビット数は、s y n c \_ p e r i o d ビット以下となる。

50

## 【 0 1 3 8 】

以下、このような出力符号列 2 0 5 を生成する図 1 9 の出力符号化装置 2 0 0 の構成と動作について詳細に説明する。

## 【 0 1 3 9 】

カウンタ 1 7 0 1 は、多重化器 1 1 1 から同期符号挿入要求信号 2 0 3 が入力され、多重化符号列 2 0 1 として同期符号 3 0 1 の最初のビットが入力されると“ 1 ”にセットされ、同期符号 3 0 1 の全ビットが入力されると同期符号長 `sync_len` にセットされる。その後、カウンタ 1 7 0 1 は同期符号 3 0 1 の次のビットから順に次の同期符号の直前のビットが出力されるまでカウントアップしてゆく。

## 【 0 1 4 0 】

切替制御器 1 7 0 4 は、多重化符号列 2 0 1 として同期符号の最初のビットから次の同期符号の前のビットまでが入力されている時は、切替器 1 7 0 3 を多重化符号列 2 0 1 の側に切り替え、多重化符号列 2 0 1 が出力符号列 2 0 5 として出力されるように制御する。

## 【 0 1 4 1 】

そして、1 フレームの最後の部分では、次の同期符号が同期符号挿入位置に挿入されるようにするためのビット挿入（ビットスタッフィング）を行う。多重化器 1 1 1 は、1 フレームの多重化符号列 2 0 1 の出力が終了すると、次のフレームの同期符号挿入要求信号 2 0 3 を出力する。これを受けて切替制御器 1 7 0 4 は、切替器 1 7 0 3 をスタッフィングビット発生器 1 7 0 5 の側に切り替え、スタッフィングビット 1 2 2 3 を出力符号列 2 0 5 として出力する。このスタッフィングビット 1 2 2 3 は全ビットを“ 1 ”としてもよいし、“ 0 ”としてもよいし、“ 0 1 0 1 ... ”のような特定のパターンとしてもよい。

## 【 0 1 4 2 】

次に、本実施形態に係る動画像復号化装置について説明する。

図 2 1 は、図 1 8 の動画像符号化装置に対応した動画像復号化装置の構成を示すブロック図である。図 8 と相対応する部分に同一符号を付して第 1 の実施形態との相違点を中心に説明すると、本実施形態では入力符号化装置 8 0 0 の構成および動作が異なる。また、入力復号化装置 8 0 0 から逆多重器 8 1 1 に入力される信号は符号列 8 0 1 および同期符号検出信号 8 0 3 のみであり、逆多重器 8 1 1 から入力復号化装置 8 0 0 に入力される信号が無い。

## 【 0 1 4 3 】

図 2 2 は、入力復号化装置 8 0 0 の構成を示すブロック図である。この入力復号化装置 8 0 0 は、入力符号列 2 0 5 中の同期符号を検出する同期検出器 1 9 0 1 と、入力符号列 2 0 5 のビット数を計数するカウンタ 1 9 0 2 から構成される。

## 【 0 1 4 4 】

カウンタ 1 9 0 2 は、復号化の最初の段階では“ 0 ”にリセットされており、入力符号列 2 0 5 が 1 ビット入力される度に計数値 1 9 1 1 を“ 1 ”ずつカウントアップしていく。

## 【 0 1 4 5 】

同期検出器 1 9 0 1 では、カウンタ 1 9 0 2 の計数値 1 9 1 1 に基づいて同期符号挿入位置でのみ同期符号の検出を行う。例えば、同期符号挿入間隔を `sync_period`、計数値 1 9 1 1 を `bit_count` とし、同期符号の長さを `sync_len` とすれば、

$0 < \text{bit\_count} \bmod \text{sync\_period} < \text{sync\_len}$   
 の時のみ同期検出を行う。ここで、 $A \bmod B$  は A を B で除したときの剰余を表す。同期検出器 1 9 0 1 は、同期符号を検出すると同期符号検出信号 8 0 3 を出力する。

## 【 0 1 4 6 】

入力復号化装置からの符号列 8 0 1 は、入力符号列 2 0 5 がそのまま出力され、逆多重化器 8 1 1 に入力される。以降、図 2 1 の動画像復号化装置と同様に逆多重化、復号化の処理が行われる。

## 【 0 1 4 7 】

10

20

30

40

50

フレームの最後のスタッフィングビットSTUFFを予め定められたビットパターンとした場合、逆多重化器811においてこのスタッフィングビットSTUFFが予め定められたパターンと一致するか判定し、一致しなかった場合には入力符号列205中に誤りがあったと判定して、第1の実施形態で示した動画像符号化装置で述べたような画質劣化が大きくなるようにするための処理を行ってもよい。

#### 【0148】

(第4の実施形態)

次に、本発明に係る第4の実施形態について説明する。

本実施形態に係る動画像符号化装置はその全体構成は図18の動画像符号化装置と同一であり、出力符号化装置の動作が第3の実施形態と異なる。

10

#### 【0149】

図23は、図18中の出力符号化装置200の構成を示すブロック図である。図19の出力符号化装置と相対応する部分に同一の符号を付してその差異を中心に説明すると、疑似同期符号を防止するためのビットスタッフィング処理を行うビット挿入器1211が追加されている。

#### 【0150】

ビット挿入器1211において、多重化符号列201に対して疑似同期の発生の防ぐためのビット挿入が行われる。これは、出力符号列205中に同期符号と同一のビットパターンがあると同期符号を一意に復号できなくなるため、これを防ぐための処理である。例えば、同期符号が図5に示すようにsync\_\_0\_\_len ビットの“0”、1ビットの“1”および同期符号の種類を示すsync\_\_nb\_\_len ビットの“xxxxx”からなる場合、同期符号以外の符号列中にsync\_\_0\_\_len ビット以上“0”が連続しないように“1”を挿入すれば疑似同期の発生を防ぐことができる。

20

#### 【0151】

同期符号は同期符号挿入装置にしか挿入されない。従って、疑似同期発生防止のためのビット挿入操作は同期符号挿入位置だけで行えばよい。そこで、出力符号列205の総ビット数を示す計数値1221に基づいてビット挿入が必要であるかを判定する。計数値1221をtotal\_\_len とすれば、

$$0 < \text{total\_len} \_ \text{mod} \_ \text{sync\_period} \text{ sync\_0\_len}$$

30

となる区間で多重化符号列201中の“1”の数を計数し、この区間に“1”が全く無くなければ1ビットの“1”を挿入する。ここで、A mod BはAをBで除したときの剰余を表す。

#### 【0152】

また、誤りによる同期符号の誤検出の確率を低くするためには、以下のようにビット挿入を行えばよい。

#### 【0153】

同期符号にnビット誤りが混入した場合にも同期符号を検出するためには、後述する動画像符号化装置の入力符号化装置で真の同期符号とハミング距離がn以下の符号語は同期符号と判定する必要がある。しかし、同期符号以外の符号列をそのままにしてこのような判定を行うと、同期符号以外の符号列中にも同期符号とハミング距離がn以下のビットパターンが存在することがあり、これが同期符号挿入位置にあると誤って同期符号と判定されてしまう。

40

#### 【0154】

そこで、ビット挿入器211で多重化符号列201に以下のようにビット挿入を行うことによって、多重化符号列201中の同期符号挿入位置にある同期符号以外の符号列を同期符号とのハミング距離が $2 \times n + 1$ 以上離れるように変換する。具体的には、

$$0 < \text{total\_len} \_ \text{mod} \_ \text{sync\_period} \text{ sync\_0\_len} - (2 \times n + 1)$$

となる区間での“1”の数(=n0とする)を計数し、n0が $2 \times n + 1$ 未満であつたら

50

$2 \times n + 1 - n_0$  ビットの “ 1 ” を多重化符号列 2 0 1 に挿入する。

【 0 1 5 5 】

こうしてビット挿入が行われた後の符号列 1 2 2 2 は、図 1 9 の出力符号化装置と同様にフレームの最後の区間のビット挿入（図 2 0 中 S T U F F ）が行われ、出力符号列 2 0 5 として出力される。

【 0 1 5 6 】

次に、本実施形態に係る動画像復号化装置について説明する。動画像復号化装置の全体構成は図 2 1 の動画像復号化装置と同一であり、入力符号化装置 8 0 0 の動作が第 3 の実施形態と異なる。

【 0 1 5 7 】

図 2 4 は入力復号化装置 8 0 0 の構成を示すブロック図である。図 2 2 の入力復号化装置と相対応する部分に同一の符号を付して第 3 の実施形態との差異を中心に説明するビット除去器 1 9 0 5 が追加されている。

【 0 1 5 8 】

入力符号列 2 0 5 は、挿入ビット除去器 1 9 0 5 に入力され、図 2 3 の出力符号化装置のビット挿入器 1 2 1 1 で挿入された疑似同期符号を防ぐための挿入ビットを取り除く処理を行う。前述のように、ビット挿入は同期符号挿入位置でのみ行われているため、計数器 1 9 0 2 の計数値 1 9 1 1 に基づいて同期符号挿入位置を判定する。

【 0 1 5 9 】

例えば、同期符号が図 5 に示す符号語であるものとし、ビット挿入器 1 2 1 1 において同期符号とのハミング距離が  $2 \times n + 1$  以上離れるように同期符号の最初の “ 0 0 0 0 ... ” の部分にビット挿入が行われている場合には、同期符号挿入位置から  $\text{sync\_0\_1\_len} - (2 \times n + 1)$  ビットの “ 1 ” の数を計数し、 $n_0$  が  $2 \times n + 1$  未満であったら  $2 \times n + 1 - n_0$  ビット削除する。

【 0 1 6 0 】

ここで、挿入ビットは “ 1 ” と定められているため、挿入ビットと判定されたビットが “ 0 ” となっていたときは、同期符号挿入区間に誤りが混入したと考えられる。この場合は、図示しない誤り検出信号を逆多重化器 8 1 1 に出力し、第 1 の実施形態で述べたのと同様に、再生画像に大きな劣化が生じないように処理を行ってもよい。

【 0 1 6 1 】

図 2 3 のビット挿入器 1 2 1 1 でのビット挿入処理は、同期符号以外の全ての同期符号挿入区間にあらかじめ定められたビット数の挿入ビットを挿入するようにしてもよい。2 5 はこのようなビット挿入処理を行ったときの出力符号列 2 0 5 の一例を示したものである。図中 S B が挿入されたビットを示している。

【 0 1 6 2 】

例えば、同期符号が図 5 に示すように  $\text{sync\_0\_1\_len}$  ビットの “ 0 ”、1 ビットの “ 1 ” および同期符号の種類を示す  $\text{sync\_nb\_len}$  ビットの “ x x x x x ” からなる場合、同期符号挿入区間の最初から  $\text{sync\_0\_1\_len}$  ビットの区間の予め定められた位置に 1 ビットの挿入ビット S B を挿入する。

【 0 1 6 3 】

挿入ビット S B は、常に “ 1 ” としてもよい。また、同期符号挿入区間の最初から  $\text{sync\_0\_1\_len}$  ビットの区間のビットパターンに応じて、その区間の “ 1 ” の数が 1 個以上となるように適応的に挿入ビット S B を決定してもよい。

【 0 1 6 4 】

さらに、挿入ビット S B を同期符号挿入区間の最初から  $\text{sync\_0\_1\_len}$  ビットの区間の奇パリティとすることにより、同期符号と同一のビットパターンの出現を避けると共に、このビットパターンに混入した誤りの検出をも行うことが可能である。

【 0 1 6 5 】

図 2 5 ( b ) は、このようなビット挿入処理を行った出力符号列を示している。この例では、同期符号挿入位置から最初の部分に 1 ビットの挿入ビット S B を挿入している。この

10

20

30

40

50

挿入ビットSBは、これと次のビットから  $\text{sync\_0\_len} - 1$  ビットを合わせた区間の“1”の数が必ず奇数となるように決定する。例えば、図25(b)の左側の例では、挿入ビットSBは“1”となる。また、図25(b)の右側の例では、挿入ビットSBの次のビットから  $\text{sync\_0\_len} - 1$  ビットの区間が全て“0”であっても挿入ビットSBが“1”になるため、同期符号挿入区間に必ず1ビット以上の“1”が入ることになり、同期符号と同一のビットパターンは生じない。また、挿入ビットSBはパリティチェックビットとしても働くため、この区間に混入したビット誤りを検出することが可能である。

#### 【0166】

また、挿入ビットSBを次の同期符号挿入位置までの全ビットの奇パリティチェックビットとしてもよい。但し、挿入ビットのSB次のビットから  $\text{sync\_0\_len} - 1$  ビットが全て“0”の場合のみ同期符号と同一のビットパターンの出現を避けるために、挿入ビットSBを必ず“1”とする。これにより、全ビットのパリティチェックによる誤り検出を行うことが可能である。

10

#### 【0167】

誤りによる同期符号の誤検出の確率を低くするためには、より多くのビット数を挿入することが望ましい。例えば、nビットの誤りが入っても同期を正しく検出するためには、この区間の予め定められた位置に  $2 \times n + 1$  ビットの“1”を挿入する。

#### 【0168】

本実施形態では、上述のビット挿入器1211の動作に対応して、図24のビット除去器1905の動作も異なったものになる。すなわち、ビット除去器1905ではビット挿入器1211でビット挿入を行った予め定められた位置の挿入ビットSBを削除する処理を行う。

20

#### 【0169】

ここで、挿入ビットSBを常に“1”とした場合には、入力符号列205中のビット挿入位置にあるビットが“0”であったときビット誤りと判定して、誤り検出信号(図示しない)を逆多重化器811に出力し、復号画像に大きな劣化を生じないようにするための処理を行ってもよい。

#### 【0170】

第1～第4の実施形態においては、多重化器111における多重化は図2のように予測モード情報303、動きベクトル情報304、および、残差DCT計数305を符号化フレームを単位としてまとめて多重化する例を示したが、図26のように符号化領域(例えばマクロブロック、GOB等)単位に予測モード情報303、動きベクトル情報304、残差DCT計数305をまとめるように多重化してもよい。この場合、ピクチャヘッダ302と、それ以外の情報とで異なる誤り訂正/検出符号を用いてもよいし、同一の誤り訂正/検出符号を用いてもよい。あるいは、ピクチャヘッダのみに誤り訂正/検出符号を用いてもよいし、各フレームの予め定められたビット数の一部の符号列に誤り訂正/検出符号を用いてもよいし、誤り訂正/検出符号を全く用いなくともよい。

30

#### 【0171】

また、フレーム(ピクチャ)単位だけでなく、フレーム内の部分領域や、複数のフレームをまとめたレイヤ単位で多重化を行い、これらの多重化単位(レイヤ単位)毎に同期符号を挿入してもよい。

40

#### 【0172】

図28は、このような多重化の例を示したものである。この図28の例では、複数の符号化ブロックをまとめたマクロブロック、複数のマクロブロックをまとめたGOB、ピクチャ(フレーム)、複数のピクチャをまとめたセッションの4つのレイヤ毎に多重化を行う。これらのうち、セッション、ピクチャ、GOBの各レイヤは、それぞれの同期符号(図中SSC, SEC, PSC, GSC)を用いる。SSC, SEC, PSC, GSCにはそれぞれ異なる符号を用い、どのレイヤの同期符号が検出されたかを区別できるようにする。図5のような同期符号を用いる場合、同期符号の種類を示す  $\text{sync\_nb\_len}$

50

ビットの部分でこれらの同期符号の区別を行えばよい。

#### 【0173】

このような多重化を行った場合も、セッション、ピクチャ、GOBの同期符号の一部ないしは全てに対して前述の実施形態におけるフレーム同期符号と同様の処理を行ってもよい。図29は、このような処理を行った出力符号列の一例を示したものである。同図に示されるように、PSC、GSCの前にスタッフィングビットSTUFFが挿入されており、SSC、PSC、GSCは図中矢印の同期符号挿入位置に挿入されている。このため、前述の実施形態でフレーム同期符号PSCについて説明したのと同様に、各同期符号の検出精度が向上する。

#### 【0174】

図15、図16および図17のフレーム長情報POINTERと同様の長さ情報をセッション、ピクチャ、GOBの各同期符号に付加することも可能である。図15および図16のように、フレーム長情報POINTERに対して誤り訂正/検出符号による保護を行っている場合、フレーム長情報POINTERのみでなく、同期符号の種類を示すsync\_\_nb\_\_len ビットの部分に対しても併せて誤り訂正/検出符号を用いることにより、同期符号の位置のみでなく、その種類も正しく検出できる確率が向上する。また、セッション、ピクチャ、GOBのヘッダ情報(図中SH, PH, GF)の一部または全部も併せて誤り訂正/検出符号を用いて保護を行ってもよく、これによって各ヘッダ情報の誤り耐性も向上する。

#### 【0175】

本実施形態のように疑似同期符号を防止するためのスタッフィング処理を行う場合、以下のような処理を行って同期符号挿入間隔sync\_\_periodを同期符号の長さ以下にすることも可能である。

#### 【0176】

まず、動画像符号化装置の出力符号化装置における処理について説明する。ここで、同期符号は図5に示すようにsync\_\_0\_\_len ビットの“0”と1ビットの“1”からなる符号語とする。図23の出力符号化装置において、ビット挿入器1211から出力したビット数を示す計数值1221をtotal\_\_len とすれば、total\_\_len を同期符号挿入間隔sync\_\_periodで除した剰余が、同期符号の最初の“0”のビット数sync\_\_0\_\_len から1を引いた値をsync\_\_periodで除した剰余と一致したとき、すなわち、

$$\text{total\_len} \bmod \text{sync\_period} = (\text{sync\_0\_len} - 1) \bmod \text{sync\_period} \quad (1)$$

となったとき、そのときの出力ビットから(sync\_\_0\_\_len - 1)ビット前までの出力ビット中の“1”の数(n1とする)を計数し、“1”が全くなかったら(すなわち、n1 = 0であったら)1ビットの“1”を挿入する。

#### 【0177】

図33(a)に、このような処理を行った出力符号列の例を示す。図中、下向きの矢印が同期符号挿入位置を示しており、同期符号は23ビットの“0”(すなわち、sync\_\_0\_\_len = 23)と、1ビットの“1”からなる。図の例では同期符号挿入間隔sync\_\_period は同期符号の長さ(=24ビット)よりも短い8である。

#### 【0178】

図中、区間1~4は前述のn1を計数する区間を示している。各区間で順次“1”の数n1を計数し、n1 = 0であったらば、その区間の次のビットにスタッフィングビットを挿入する。区間1ではn1 > 0なので、スタッフィングの必要はない。区間2ではn1 = 0となるので、この区間の次に1ビットのスタッフィングビット3301を挿入する。区間3ではスタッフィング3301が挿入されたことにより、n1 = 1となるため、スタッフィングビットを挿入する必要はない。

#### 【0179】

以上のようなビットスタッフィング処理を行うことにより、出力符号列中の同期符号以外

10

20

30

40

50

の部分で同期符号と同一のビットパターンが無くなるため、疑似同期を生じることが無くなる。

#### 【0180】

一方、伝送路誤りによる同期符号の誤検出の確率を低くするためには、以下のようにビット挿入を行えばよい。

#### 【0181】

同期符号に  $n$  ビット誤りが混入した場合にも、同期符号を正しく検出するためには、ビット挿入 1211 において、出力符号列中の同期符号以外の部分と同期符号とのハミング距離が  $2 \times n + 1$  以上になるようにビット挿入処理を行えばよい。

#### 【0182】

これは、図 23 における出力符号列 205 の総ビット数を示す計数値 1221 total\_\_len を同期符号挿入間隔 sync\_\_period で除した剰余が、同期符号の最初の “0” のビット数 sync\_\_0\_\_len から  $2 \times n + 1$  を引いた値を sync\_\_period で除した剰余と一致したとき、すなわち、

$$\text{total\_len} \bmod \text{sync\_period} = (\text{sync\_0\_len} - (2 \times n + 1)) \bmod \text{sync\_period} \quad (2)$$

となったとき、そのときの出力ビットから (sync\_\_0\_\_len - ( $2 \times n + 1$ )) ビット前までの出力ビット中の “1” の数 ( $n1$  とする) を計数し、“1” の数が ( $2 \times n + 1$ ) 未満、すなわち

$$n1 < 2 \times n + 1$$

であったら、( $2 \times n + 1 - n1$ ) ビット “1” を挿入する。

#### 【0183】

図 5 に示したような複数ビットの “0” から始まる同期符号を用いる場合、同期符号の直前のビット列中の “1” の数が少ないとこの部分で同期誤検出が起こることがある。これを防ぐために、同期符号からその直前の同期符号挿入位置までの sync\_\_period ビットの区間の “1” のビット数が  $2 \times n + 1$  ビット以上になるように、フレームの最後の区間のビット挿入 (図 20 中、STUFF) を出力するようにしても良い。

#### 【0184】

これには、必ず  $2 \times n + 1$  ビット以上の “1” が含まれる STUFF を用いてもよいし、あるいは出力符号列に応じて STUFF を決定するようにしてもよい。すなわち、STUFF を含めた出力符号列中の、同期符号の直前の sync\_\_period ビットの “1” のビット数が  $2 \times n + 1$  ビット以上になるように、STUFF を決定してもよい。

#### 【0185】

図 33 (b) に、このような処理を行った出力符号列の例を示す。図中、区間 1 ~ 4 は前述の  $n1$  を計数する区間を示している。各区間で順次 “1” の数を  $n1$  を計数し、 $n1 < 2 \times n + 1$  であったらその区間の次のビットにスタッフィングビットを挿入する。区間 2 では  $n1 = 1$  となるので、この区間の次に ( $2 \times n + 1$ ) - 1 = 2 ビットのスタッフィングビット 3311 を挿入する。区間 3 ではスタッフィング 3311 が挿入されたことにより  $n1 = 3$  となるため、スタッフィングビットを挿入する必要は無い。

#### 【0186】

さらに、同期符号の直前の部分での同期符号誤検出を防ぐために、以下のように STUFF を決定する。STUFF の直前のビットを 3312 とする。このビット 3312 の直前の同期符号挿入位置から直後の同期符号挿入位置までの sync\_\_period ビット間 (区間 5) の “1” の数は 1 ビットしかないため、図 5 のような複数の “0” が連続する同期符号を用いる場合にはこの部分で同期誤検出を生じる可能性がある。このため、同期符号を挿入する位置を次の同期符号挿入位置までずらし、多くの “1” を含む STUFF 3313 を出力する。これにより、同期符号の直前の sync\_\_period ビット区間 (区間 6) には  $2 \times n + 1$  ビット以上の “1” が含まれるようになるため、同期誤検出を防止することができる。

10

20

30

40

50

## 【0187】

以上のようなビットスタッフィング処理を行うことにより、出力符号列中の同期符号以外の部分で同期符号とのハミング距離を  $2 \times n + 1$  以上にすることができるため、同期誤検出の確率が減少する。

## 【0188】

次に、動画像復号化装置の入力復号化装置における処理について説明する。図24のビット除去器1905において、入力した符号列のビット数を示す計数値1905を `total_len` とすれば、`total_len` が1の条件を満たす値になった時点で、そのときの入力ビットから (`sync_0_len - 1`) ビット前までの入力ビット中の“1”の数 (`n1` とする) を計数し、“1”が全くなかったら、すなわち、`n1 = 0` であつたら、1ビット削除する。

10

## 【0189】

同期符号に `n` ビット誤りが混入した場合にも同期符号を検出するために、ビット挿入器1211で出力符号列中の同期符号以外の部分と同期符号とのハミング距離  $2 \times n + 1$  以上になるようにビット挿入処理を行った場合には、次のような処理を行えばよい。`total_len` が式(2)を満たす値になった時点で、そのときの入力ビットから (`sync_0_len - (2 \times n + 1)`) ビット前までの出力ビット中の“1”の数 (`n1` とする) を計数し、“1”の数が ( $2 \times n + 1$ ) 未満、すなわち  $n1 < (2 \times n + 1)$

であつたら ( $2 \times n + 1 - n1$ ) ビット削除する。

20

## 【0190】

出力符号化装置および入力復号化装置において以上のような処理を行って、同期符号挿入間隔 `sync_period` を同期符号の長さ以下の短いビット数とすることにより、スタッフィングビット `STUFF` のビット数が削減され、符号化効率力が向上する。特に、同期符号の長さが長い場合や、多くの同期符号が挿入される場合には、このスタッフィングビット `STUFF` のビット数削減による符号化効率向上の度合いが大きい。例えば、動画像符号化における `GOB` / スライスのように、画面を1ないし複数のマクロブロック、あるいはマクロブロックライン毎に区切り、その単位毎に同期符号を挿入する方式においては、多くの同期符号が挿入されるため、`STUFF` のビット数削減による符号化効率の度合いがより大きくなる。

30

## 【0191】

また、図28のように複数のレイヤ構造の多重化を行う場合には、レイヤに応じて異なる長さの同期符号を用いるようにしても良い。

## 【0192】

図34(a)は、このような同期符号の例を示したものである。4種類の同期符号のうち、`SSC`、`SEC`、`PSC` はいずれも23ビットの“0”、1ビットの“1”、および同期符号の種類を示す8ビットの合計32ビットからなる。一方、`GOB` レイヤの同期符号 `GSC` は、16ビットの“0”と1ビットの“1”からなる17ビットの同期符号であり、他の同期符号より短い符号語である。

## 【0193】

このように `GSC` のみを短い符号語としたのは、`GOB` は1ないし複数のマクロブロック (`MB`) からなる、画面内を小領域に区切った符号化単位であることから、一般に `GOB` レイヤの同期符号は他の同期符号に比べて数が多く、同期符号語長を短くすることで、出力符号列の符号量を削減することができるためである。また、このようにすると、同じ符号量ならばより多くの `GSC` を出力することができ、画面内をより細かな `GOB` 領域に区切って符号化することが可能になるため、伝送路誤りが発生したときの再生画像の品質が向上する。

40

## 【0194】

第4の実施形態で説明したような疑似同期を防止するための処理、すなわち、同期符号以外の符号列中に同期符号と同一のビットパターンを生じないようにするビットスタッフィ

50



ング処理を行うようにしてもよい。伝送路誤りによる同期符号の誤検出の確率を低くするためのビットスタッフィング処理、例えば、ビット長の長い同期符号（図34(a)の例ではSSC, SEC, PSC）と同一のビットパターンが生じないことが保証されているビット列に対しても、最も短い同期符号（図34(a)の例ではGSC）と同一のパターンが生じないようにビットスタッフィング処理を行えば、全ての同期符号と同一のビットパターンが生じないようにすることができる。この処理は、全レイヤの符号列に対して行ってもよいし、最も短い符号を用いるレイヤより下位のレイヤ（図の例ではGOBレイヤ、マクロブロックレイヤ）の符号列に対して行ってもよいし、そのレイヤよりも一つ上のレイヤ以下のレイヤ（ピクチャレイヤ、GOBレイヤ、マクロブロックレイヤ）の符号列に対して行ってもよい。さらに、この処理を予め定められたレイヤの符号列に対してのみ行ってもよい。

10

#### 【0195】

伝送路誤りが生じて長さの異なる同期符号同士を識別しやすくするために、同期符号語およびその前後の処理を以下のようにしてもよい。

#### 【0196】

(i) 複数ビットの“0”と、それに続く“1”からなる同期符号を用いる場合には、長い符号語と短い符号語で、この“1”の同期符号挿入位置から見た相対位置が異なるようにしてもよい。図34(b)の例では、PSCの“1”3411と、GSCの“1”3412は異なる位置にあり、別の同期符号中の同位量のビット（3411に対する3413、および3412に対する3414）はいずれも“0”である。このようにする事により同期符号およびその部分列のハミング距離が大きくなるため、伝送路誤りが生じて異なる同期符号同士を識別しやすくなる。

20

#### 【0197】

(ii) 長さの短い同期符号の前にスタッフィングビットを挿入しても良い。例えば、図中3401のように長さの短いGSCの前に1ないし複数の“1”からなるスタッフィングビット3401を挿入することにより、OSCと他の同期符号の部分列とのハミング距離を大きくすることができる。

#### 【0198】

(iii) 長さの短い同期符号の後ろにスタッフィングビットを挿入してもよい。例えば、長い同期符号中の同期符号の種類を識別する部分とのハミング距離が大きくなるように、GSCの後ろにビット挿入3402を行ってもよい。

30

#### 【0199】

##### (第5の実施形態)

次に、本発明に係る第5の実施形態について説明する。

本実施形態による動画像符号化装置および動画像復号化装置の全体の構成は第1の実施形態と同様であり、出力符号化装置200および入力復号化装置800の1同期区間の最初および最後の部分での処理がこれまでの実施形態と異なっている。

#### 【0200】

図27(a)(b)(c)は、本実施形態における動画像符号化装置からの出力符号列205の一例を示す図である。この出力符号列205は、同期符号PSCの後に前フレーム（フレームn-1）の符号列の一部2701が入っており、この符号列2701と現フレーム（nフレーム）の符号列との境界2703（現フレームの符号列のスタートポイント）、言い換えれば多重化符号列の境界を示すポイント情報2702(SA)があり、これに伴い1フレームの最後のスタッフィングビット（図4中STUFF）が無くなっていることが図4の出力符号列と異なっている。

40

#### 【0201】

動画像符号化装置内の出力符号化装置200においては、各同期符号挿入位置においてそのフレームの残りの符号列のビット数 `resid__bit` をチェックする。`resid__bit` と同期符号PSCとポイント情報SAのビット数の和が同期符号挿入間隔 `sync__period` ビットよりも少ない場合には、そのフレームの残りの符号列を出

50

力符号列 2 0 5 中に出力する前に、同期符号 P S C を出力する。次に、ポインタ情報 S A (この場合、 r e s i d \_ b i t を表す) を出力し、その後に残りの符号列 2 7 0 1 を出力する。その後、次のフレームの符号列を出力していく。

【 0 2 0 2 】

動画像復号化装置内の入力符号化装置 8 0 0 においては、各同期符号挿入位置において同期符号の検出を行い、同期符号が検出された場合にはその後ろにポインタ情報 S A とフレームの残りの情報が続いているものとして処理を行う。

【 0 2 0 3 】

図 2 7 中のフレーム n - 1 とフレーム n の境界部を例に説明すると、同期符号 P S C の直前 2 7 0 4 までの復号処理を終えた後、その後ろの同期符号挿入区間で同期符号を検出し、同期符号が検出された場合には、次にポインタ情報 2 7 0 2 の復号化し、フレーム n - 1 の符号列があと何ビットあるかを求める。これを基にして、ポインタ情報の直後の符号列からポインタ情報で示されるビット数を取り出し(図 2 7 中 2 7 0 3 まで)、これが 2 7 0 4 に続くものとして符号列 8 0 1 を出力する。その後ろの符号列(図 2 7 中 2 7 0 3 から)は次のフレーム、すなわちフレーム n として処理を行う。

【 0 2 0 4 】

本実施形態においては、図 2 7 ( a ) に示すように出力符号列の一部または全てに対して誤り訂正 / 検出符号化を行ってもよい。この場合、誤り訂正 / 検出符号の種類は全て同じでもよいし、異なるものを用いてもよい。

【 0 2 0 5 】

また、図 2 7 ( b ) に示すように、誤り訂正 / 検出符号化を行わないようにしてもよい。

【 0 2 0 6 】

さらに、図 2 7 ( c ) のように、図 1 5 や図 1 6 に示したような 1 フレームの符号列ビット数を示すフレーム長情報 P O I N T E R を挿入してもよい。この場合、フレーム長情報 P O I N T E R は、そのフレームの同期符号 P S C から次のフレームの同期符号 P S C までのビット数を表すようにしてもよい。

【 0 2 0 7 】

図 2 7 ( a ) のように誤り訂正 / 検出符号化を行う場合、同期符号 P S C から次の検査ビット C H K までの部分の誤り訂正 / 検出符号化は、ポインタ情報 S A 、フレーム n - 1 の残りの符号列 2 7 0 1 および、2 7 0 3 以降のフレーム n の符号列をまとめてひとつの情報ビットとして誤り訂正 / 検出符号を行う。

【 0 2 0 8 】

ポインタ情報 S A は誤り訂正 / 検出符号化を行った情報としてもよい。この場合、同期符号 P S C (あるいはその一部)、フレーム長情報 P O I N T E R とポインタ情報を合わせて誤り訂正 / 検出符号化を行うようにしてもよい。

【 0 2 0 9 】

次に、スタッフィングビット S T U F F の具体例を説明する。

【 0 2 1 0 】

図 3 0 ( a ) ( b ) は、上述したスタッフィングビット S T U F F の具体例としてのスタッフィングビット S T U F F の符号表の例を示した図である。図 3 0 ( a ) ( b ) のいずれも、出力符号列の逆方向から一意に復号できることが特徴であり、これによってスタッフィングビット S T U F F の開始位置を一意に特定できる。従って、スタッフィングビット S T U F F の直前の符号列の復号終了位置と、スタッフィングビット S T U F F の開始位置を比較することにより、符号列中に混入した誤りを検出できると共に、同期符号から逆方向に復号を行う符号化方式に用いた場合に逆方向復号の開始地点を特定することができる。

【 0 2 1 1 】

さらに、図 3 0 ( a ) ( b ) の符号表に示すスタッフィングビット S T U F F は、その最初のビットが必ず “ 0 ” になっており、後述のような簡易的な復号による誤り検出が可能である。

10

20

30

40

50

## 【0212】

図31は、図30(a)(b)の符号表に示したスタフニングビットSTUFFを含む符号列の復号処理の例を示したものである。図31では、同期符号挿入位置の直前のスタフニングビットの例を示しているが、これ以外の任意の同期符号挿入位置の直前にスタフニングビットビットを挿入して同様の処理を行うことも可能である。図31中、矢印3101～3103は順方向に復号化を行った場合のスタフニングビットSTUFFの直前の符号列(“xxx...”で示す)の復号終了位置の例であり、矢印の右端が復号終了位置を表す。符号列に誤りが混入しておらず、正常に復号化が行われた場合には、矢印3101のようにスタフニングビットSTUFFの直前の符号列の復号終了位置とスタフニングビットSTUFFの開始位置は一致する。

10

## 【0213】

一方、符号列に誤りが混入した場合には、スタフニングビットSTUFFの直前の符号列の復号終了位置は矢印3102、3103のようにスタフニングビットSTUFFの開始位置とずれる。このような場合には、符号列に誤りがあったと判定する。

## 【0214】

復号化装置においては、スタフニングビットSTUFFの直前の符号列の復号化が終了すると、次の同期符号挿入位置までスタフニングビットSTUFFを読み込み、それが図30(a)(b)に示した符号表の符号に合致するかどうかを判定する。もし、スタフニングビットSTUFFが符号表のいずれの符号にも合致しない場合は、誤りがあったと判定する。

20

## 【0215】

スタフニングビットSTUFFと符号表との合致を判定する場合、少数のビットの誤りを許容するようにしてもよい。こうすることにより、スタフニングビットSTUFFそのものに誤りが入った場合の誤りの誤検出を低減することができる。

## 【0216】

図30(a)の符号表は、必ず“0”で始まり後続のビットが“1”である。従って、スタフニングビットSTUFFの直前の符号列の復号終了位置の次のビットが“0”であるか否かだけを判定して誤り検出を行ってもよいし、あるいは、最初の“0”と後続のいくつかの“1”のみから誤り検出を行ってもよい。このようにすると、誤り検出精度は若干低下するものの、復号化に要する処理量が削減される。このように全てのスタフニングビットSTUFFが特定のビットないしは複数ビットからなる特定のビットパターンから始まる符号表を用いる場合には、復号化処理の簡略化を図ることができる。

30

## 【0217】

さらに、図30(a)(b)の符号表に示されるスタフニングビットSTUFFは、“1”のビットを多く含んでおり、図5のように“0”が多く含まれる同期符号およびその一部分とのハミング距離が離れているため、疑似同期が発生する確率が低いという利点がある。具体的には、図30(a)の符号表ではスタフニングビットSTUFFの最初のビットのみが全て“0”で、それ以外のビットは全て“1”であるため、オール“0”である同期符号およびその一部分とのハミング距離は(スタフニングビットSTUFFの長さ-1)となる。また、図30(b)の符号表ではスタフニングビットSTUFFの最初と最後のビットのみが“0”で、それ以外のビットは全て“1”であるため、同期符号およびその一部分とのハミング距離は(スタフニングビットSTUFFの長さ-2)となる。このようにスタフニングビットSTUFFと同期符号およびその一部分とのハミング距離を所定値以上、例えば(スタフニングビットSTUFFの長さ-2)以上に選ぶことにより、符号列に誤りが混入しても疑似同期符号が発生しにくい。

40

## 【0218】

この効果について、図32を用いて説明する。図32(a-0)、(b-0)はそれぞれ通常のスタフニングビット(全てのビットが“0”)および図30(a)の符号表に示すスタフニングビットSTUFFを用いたときの符号列の例を示したものであり、(a-1)、(b-1)はそれぞれ(a-0)、(b-0)に1ビットの誤りが混入したとき

50

の例を示したものである。図 3 2 ( a - 1 ) から分かるように、全てのビットが “ 0 ” である通常のスタッフィングビットでは 1 ビットの誤りが混入しただけで、( a - 1 ) の破線で示すように同期符号と同一のビットパターンが生じるため、疑似同期が発生してしまう。これに対し、図 3 0 ( a ) の符号表に示すスタッフィングビット S T U F F は、図 3 2 ( b - 2 ) に示すように誤りが混入しても同期符号と同一のパターンにならないため、疑似同期が発生しない。

#### 【 0 2 1 9 】

このように本実施形態によるスタッフィングビットは、符号列の誤りを容易に検出できると共に、符号列に誤りが混入しても疑似同期符号が発生しにくく、強い誤り耐性を持つという利点がある。

10

#### 【 0 2 2 0 】

また、本実施形態によるスタッフィングビットは逆方向から一意に復号が可能であり、その開始位置すなわちスタッフィングビットビット S T U F F の直前の符号列の終了位置を特定できるため、情報符号列が順方向からも逆方向からも復号可能な符号後の場合、図 3 1 の矢印 3 1 0 4 で示すように S T U F F の直前の符号列を逆方向から復号することができる。

#### 【 0 2 2 1 】

上述の実施形態において、スタッフィングビット S T U F F は、以下のように定めてもよい。

#### 【 0 2 2 2 】

20

( 1 ) 同期符号に図 5 に示したように s y n c \_ 0 \_ l e n ビットの “ 0 ” が含まれる場合、スタッフィングビット S T U F F の全てないしは少なくとも同期符号挿入位置の全ビットを “ 1 ” とすることにより、同期符号の “ 0 ” の部分とスタッフィングビット S T U F F のハミング距離を離すことができる。従って、このようにすることにより、スタッフィングビット S T U F F に誤りが混入して疑似同期が発生する確率を減少させることができる。

#### 【 0 2 2 3 】

( 2 ) スタッフィングビット S T U F F は、その長さを表す符号語としてもよい。復号化装置において、スタッフィングビット S T U F F 以外の符号列の復号が終了した地点より、S T U F F の長さを判定すると共に、S T U F F の復号を行って S T U F F の長さ情報を復号する。この場合、もし両者が一致しなかった場合には符号列中に誤りが混入したと判定することができる。

30

#### 【 0 2 2 4 】

また、スタッフィングビット S T U F F の符号語は、その長さを 2 進数表現したものでもよい。例えば、S T U F F が 5 ビットの場合、“ 5 ” を 2 進数表現した “ 0 0 1 0 1 ” としてもよい。あるいは、2 進数表現したものの “ 1 ” の補数や “ 2 ” の補数をとった値をスタッフィングビット S T U F F の符号語としてもよく、それにより S T U F F 中の “ 0 ” のビットの数が減少するため、( 1 ) で述べたのと同様に疑似同期の発生を抑えることができる。

#### 【 0 2 2 5 】

40

( 3 ) 順方向だけでなく、逆方向からも復号可能な符号語を用いて符号化を行った場合、復号装置においてフレームの終了地点からスタッフィングビット S T U F F を逆方向に復号して、その開始地点 ( S T U F F と他の符号語との境界地点 ) を知る必要がある。そのような場合には、例えば S T U F F を “ 0 1 1 1 1 1 1 ” のように、1 ビットないしは複数ビットの “ 0 ” で始まり残りが “ 1 ” の符号語として定めればよい。これにより、S T U F F を逆方向から復号して “ 0 ” がある位置を探せば、その地点を S T U F F の開始地点と一意に判定することができる。また、この例ではスタッフィングビット S T U F F の最初の部分以外のビットは “ 1 ” であり、( 1 ) で述べたと同様に疑似同期が発生する確率が減少する。

#### 【 0 2 2 6 】

50

(4) スタッフィングビット  $STUFF$  は出力符号列の一部または全てのビットの誤り訂正 / 検出符号の検査ビットや、パリティチェックビット等としてもよい。これにより、出力符号列中に混入したビット誤りの訂正 / 検出を行うことが可能である。

【0227】

以上の例のように、スタッフィングビット  $STUFF$  を予め定められた規則に従って生成することにより、復号装置において入力符号列中のスタッフィングビット  $STUFF$  をその生成規則と照合し、もしその生成規則に反すると判定された場合には、入力符号列中に誤りが混入したと判定することが可能である。これにより、動画復号化装置において再生画像に大きな劣化が生じないような処理を行って、入力符号列中に誤りが混入したときの再生画像の品質を向上させることができる。

10

【0228】

さらに、上記実施形態において同期符号挿入間隔  $sync\_period$  は、以下のよう

【0229】

(1) 誤り訂正 / 検出符号を用いる場合、同期符号挿入間隔  $sync\_period$  は復号化装置で同期検出を行う最小のビット数、すなわち同期符号の長さ

20

【0230】

と誤り訂正 / 検出符号を用いない場合、同期符号挿入間隔  $sync\_period$  は復号化装置で同期検出を行う最小のビット数、すなわち同期符号の長さ以上あれば十分である。フレームの最後のスタッフィングビット  $STUFF$  のビット数の平均は  $sync\_period / 2$  であるので、 $sync\_period$  をこの同期検出可能な最小ビットにとることにより、スタッフィングビット  $STUFF$  のビット数が削減され、符号化効率が向上する。

【0231】

(3) 図15、図16、図17および図27に示されるように、フレーム長情報  $POINTER$  を用いる場合には、同期符号挿入間隔  $sync\_period$  を同期符号の長さより短くしてもよい。これにより、スタッフィングビット  $STUFF$  のビット数が削減され、符号化効率が向上する。

30

【0232】

(4) 伝送路や蓄積媒体において定められた間隔のパケットやセルに区切って伝送 / 蓄積を行う場合、同期符号挿入間隔  $sync\_period$  をパケットやセルの間隔に合わせるか、その間隔の約数になるようにしてもよい。これにより、パケットやセルの先頭は必ず同期符号挿入位置になるため、パケットロスやセルロスによりパケットやセルが起きた場合にも同期符号を検出することが可能である。

【0233】

(5) 同期符号挿入間隔  $sync\_period$  は、1フレームの最小必要ビット数より短くすることが好ましい。これにより、スタッフィングビット  $STUFF$  のビット数が削減され、符号化効率が向上する。

40

【0234】

(第6の実施形態)

次に、本発明に係る第6の実施形態について説明する。

図35は、本実施形態における動画符号化装置の出力符号列の一例を示す図である。この出力符号列では、誤りによる同期符号の誤検出確率を低くするために、前述の実施例で述べたようなビット挿入処理を行っている。また、予め定められた位置、あるいは同期符号から相対的にみて予め定められた位置に、ヘッダ情報等の情報が入っている。

【0235】

50

図 3 5 の ( a ) はビット挿入処理を行う前の符号列であり、( b ) はビット挿入処理を行った後の出力符号列である。図中、斜線の部分 3 2 0 1 , 3 2 0 2 , 3 2 6 1 , 3 2 6 2 が定められた位置 ( 同期符号から相対的に見て定められた位置 ) に入れる情報であり、白抜き矢印 3 2 1 1 , 3 2 1 2 がその情報を入れる位置である。図 3 5 ( b ) 中の符号列の情報 3 2 6 1 , 3 2 6 2 が ( a ) の符号列の情報 3 2 0 1 , 3 2 0 2 にそれぞれ相当し、場合によっては符号列 ( a ) から符号列 ( b ) への変換に際して、これらの情報に対しても変換 ( すなわち、情報 3 2 0 1 から情報 3 2 6 1 への変換および情報 3 2 0 2 から情報 3 2 6 2 への変換 ) が施されることがある。

【 0 2 3 6 】

図 3 5 ( b ) 中の 3 2 0 3 は、ビット挿入処理によって挿入されたビットである。このビット挿入処理により、挿入ビットに後続するビット列は後方にずれてしまうため、定められた位置に入れる情報が定位置に入るように、その情報直前の符号列の一部を後方に移す処理を行う。例えば、情報 3 2 0 1 の一つ前の同期符号 3 2 0 5 からの挿入ビットの数の合計を  $Ns1$  とすれば、情報 3 2 0 1 の直前の図 3 5 ( a ) 中の記号 3 2 2 1 に示す  $Ns1$  ビットを情報 3 2 0 1 の直後の図 3 5 ( b ) 中の記号 3 2 3 1 の部分に移せばよい。

【 0 2 3 7 】

情報 3 2 0 1 , 3 2 0 2 中に、符号列中の特定の位置を示すポインタ等の情報が含まれている場合には、これを変換する処理を行ってもよい。具体的には、例えば情報 3 2 0 1 中に矢印 3 2 4 1 で示す位置を示す情報が含まれている場合、この位置よりも挿入ビットの数  $Ns1$  だけ後ろの矢印 3 2 5 1 で示す位置を指すように、情報 3 2 6 1 中の位置を示す情報を変換するようにする。

【 0 2 3 8 】

【 発明の効果 】

以上説明したように、本発明によれば符号化装置において複数種類の圧縮符号を多重化した多重化符号列中に予め周期的に定められた同期符号挿入位置にのみ同期符号を挿入するため、復号化装置においては、この同期符号挿入位置でのみ同期検出を行えば良く、符号列中の任意の位置に同期符号が挿入される従来方式に比べて同期符号検出の回数が削減される。

【 0 2 4 6 】

また、ビット誤りによって復号化装置に入力されるビット列が同期符号と同一のビットパターンに変化してしまうことによる疑似同期の発生の確率は、この同期符号検出回数の削減に伴って減少するため、本発明によると疑似同期の発生を少なくすることができ、同期符号検出に伴う演算処理量も削減される。

【 0 2 4 7 】

また、同期符号の誤りを考慮したビット挿入、すなわち同期符号挿入位置にあるビット列に対して同期符号とのハミング距離の変換を行うことにより、ビット列中に同期符号と同一のビットパターンが含まれなくなるため、想定したビット数以下の誤りに対しては同期符号の誤検出が生じないことが保証され、同期符号誤検出の確率が減少する。

【 図面の簡単な説明 】

【 図 1 】 本発明の第 1 および第 2 の実施形態に係る動画像符号化装置の構成を示すブロック図

【 図 2 】 図 1 の動画像符号化装置における多重化器での多重化の規則を示す図

【 図 3 】 図 1 の動画像符号化装置における出力符号化装置の構成を示すブロック図

【 図 4 】 図 1 の動画像符号化装置からの出力符号列の例を示す図

【 図 5 】 同期符号の例を示す図

【 図 6 】 図 3 の出力符号化装置における誤り訂正 / 検出切替符号化部の構成を示すブロック図

【 図 7 】 図 3 の出力符号化装置における符号列組立器の構成を示すブロック図

【 図 8 】 本発明の第 1 および第 2 の実施形態に係る動画像復号化装置の構成を示すブロック図

10

20

30

40

50

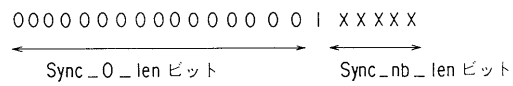
- 【図 9】図 8 の動画像復号化装置における入力復号化装置の構成を示すブロック図
- 【図 10】図 9 の入力復号化装置における符号列分解器の構成を示すブロック図
- 【図 11】従来の誤り訂正 / 検出符号切替符号化装置で得られる符号列の例を示す図
- 【図 12】本発明の第 2 の実施形態を説明するための伝送路のビット付加 / 消失によって誤りが生じた同期符号の例を示す図
- 【図 13】第 2 の実施形態における図 3 中のビット挿入器の動作を説明するための図
- 【図 14】第 2 の実施形態における図 9 中の同期検出器および挿入ビット除去器の動作を説明するための図
- 【図 15】第 1 および第 2 の実施形態においてフレーム長情報を用いて同期保護を行うようにした符号列の例を示す図 10
- 【図 16】第 1 および第 2 の実施形態においてフレーム長情報を用いて同期保護を行うようにした符号列の他の例を示す図
- 【図 17】第 1 および第 2 の実施形態においてフレーム長情報を用いて同期保護を行うようにした符号列のさらに別の例を示す図
- 【図 18】本発明の第 3 および第 4 の実施形態に係る動画像符号化装置の構成を示すブロック図
- 【図 19】本発明の第 3 の実施形態に係る動画像符号化装置の出力符号化装置の構成を示すブロック図
- 【図 20】本発明の第 3 の実施形態に係る動画像符号化装置からの出力符号列の例を示すブロック図 20
- 【図 21】本発明の第 3 および第 4 の実施形態に係る動画像復号化装置の構成を示すブロック図
- 【図 22】本発明の第 3 の実施形態に係る動画像復号化装置の入力復号化装置の構成を示すブロック図
- 【図 23】本発明の第 4 の実施形態に係る動画像符号化装置の出力符号化装置の構成を示すブロック図
- 【図 24】本発明の第 4 の実施形態に係る動画像復号化装置の入力復号化装置の構成を示すブロック図
- 【図 25】本発明の第 4 の実施形態に係る動画像符号化装置からの出力符号列の例を示すブロック図 30
- 【図 26】動画像符号化装置における多重化器での多重化の規則を示す図
- 【図 27】本発明の第 5 の実施形態に係る動画像符号化装置からの出力符号列の例を示すブロック図
- 【図 28】動画像符号化装置における多重化器での多重化の他の例を示す図
- 【図 29】図 28 で示される多重化を行った場合の各同期符号に対して処理を行った出力符号列を示す図
- 【図 30】本発明で用いるスタッフィングビットの例を説明するための符号表を示す図
- 【図 31】図 30 のスタッフィングビットを用いた場合の復号化装置の処理を説明するための図
- 【図 32】図 30 のスタッフィングビットの特徴を説明するための図 40
- 【図 33】同期符号挿入間隔を同期符号より短くした場合の出力符号列の例を示す図
- 【図 34】異なる長さの同期符号を用いる例を示す図
- 【図 35】本発明の第 6 の実施形態に係る動画像符号化装置からの出力符号列を示す図
- 【符号の説明】
- 101 ... 動き補償適応予測器
- 102 ... フレームメモリ
- 104 ... 離散コサイン変換器
- 105 ... 量子化器
- 107 ... 逆量子化器
- 108 ... 逆離散コサイン変換器 50

1 1 1 ... 多重化器	
1 3 1 ... 入力動画像信号	
2 0 0 ... 出力符号化装置	
2 0 1 ... 多重化符号列	
2 0 5 ... 出力符号列	
3 0 1 ... 同期符号	
3 0 2 ... ピクチャヘッダ	
3 0 3 ... 予測モード情報	
3 0 4 ... 動きベクトル情報	
3 0 5 ... 予測残差信号の D C T 係数	10
4 0 2 ... 同期符号挿入位置	
2 1 2 ... ビット挿入器	
2 1 1 ... 符号列組立器	
6 0 3 ... ラッチ回路	
6 0 4 ... 誤り訂正 / 検出符号化器	
7 0 1 ... カウンタ	
7 0 2 ... バッファ	
7 0 5 ... 挿入ビット発生器	
8 0 1 ... 動き補償適応予測器	
8 2 0 ... フレームメモリ	20
8 0 7 ... 逆量子化器	
8 0 8 ... 逆離散コサイン変換器	
8 1 1 ... 逆多重化器	
8 0 0 ... 入力復号化装置	
8 5 0 ... 再生画像信号	
9 0 1 , 1 9 0 1 ... 同期符号検出器	
9 0 2 , 1 9 0 2 ... カウンタ	
9 0 3 ... 符号列分解器	
9 0 4 ... 誤り訂正 / 検出符号復号化器	
9 0 5 , 1 9 0 5 ... 挿入ビット除去器	30
1 0 0 5 ... カウンタ	
1 0 0 6 ... 比較器	
1 0 1 5 ... 挿入ビット発生器	
1 1 0 1 ~ 1 1 0 4 ... ヘッダ情報	
1 2 1 1 ... ビット挿入器	

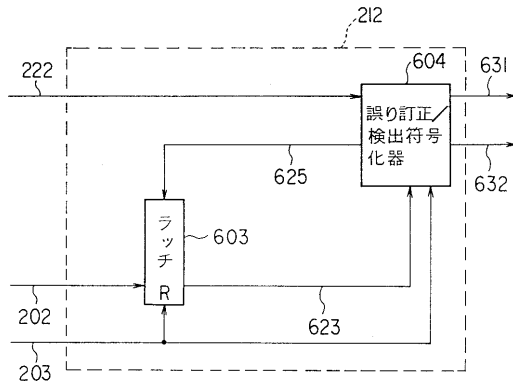




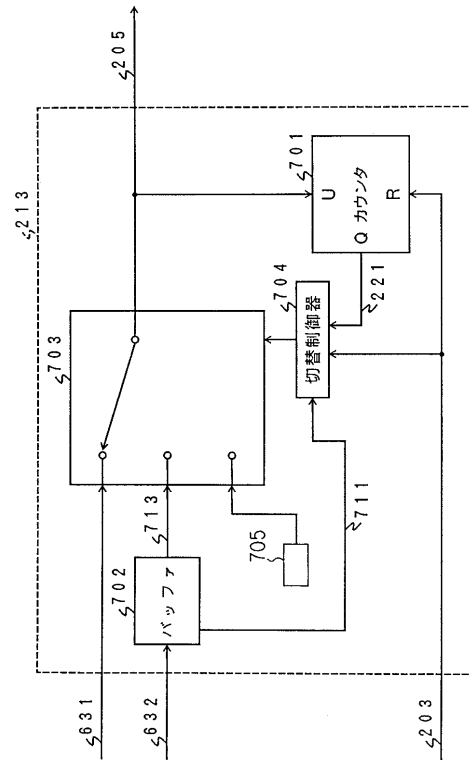
【 図 5 】



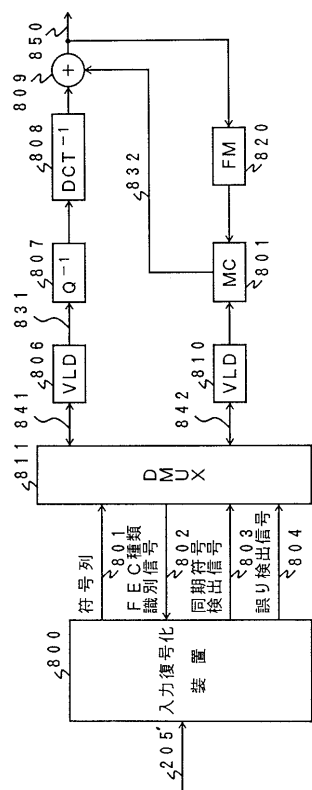
【 図 6 】



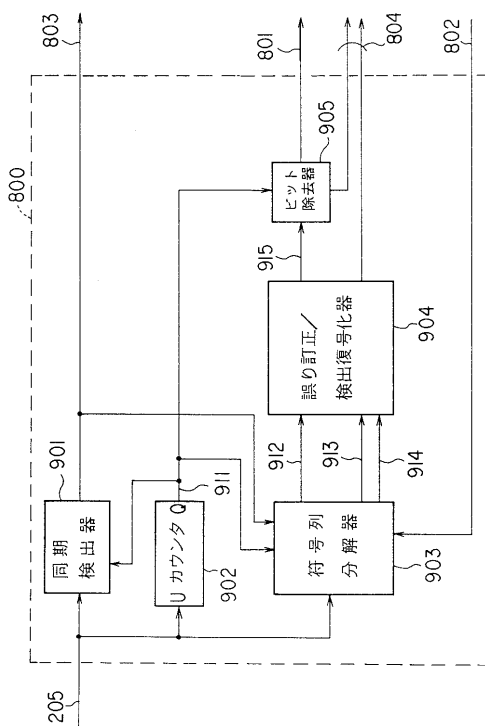
【 圖 7 】



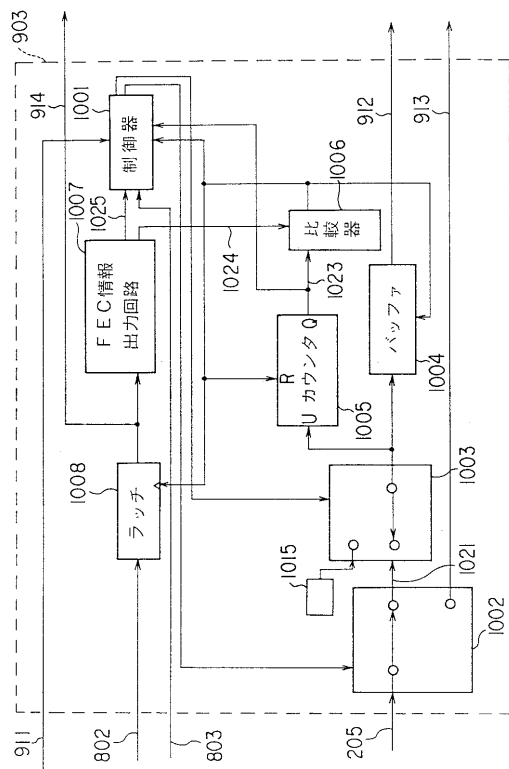
【 圖 8 】



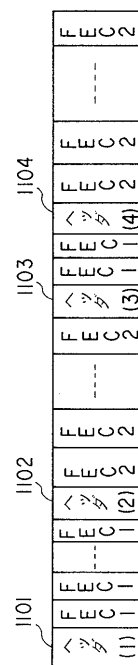
【 図 9 】



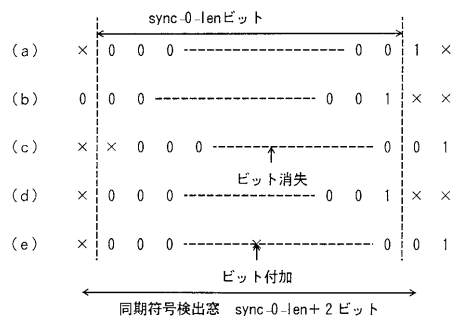
【 図 1 0 】



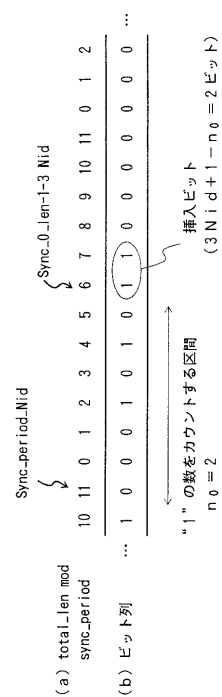
【 図 1 1 】



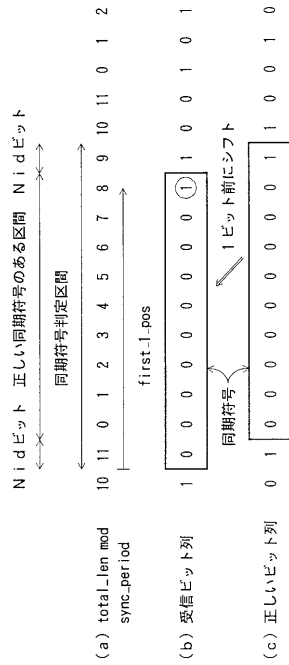
【 図 1 2 】



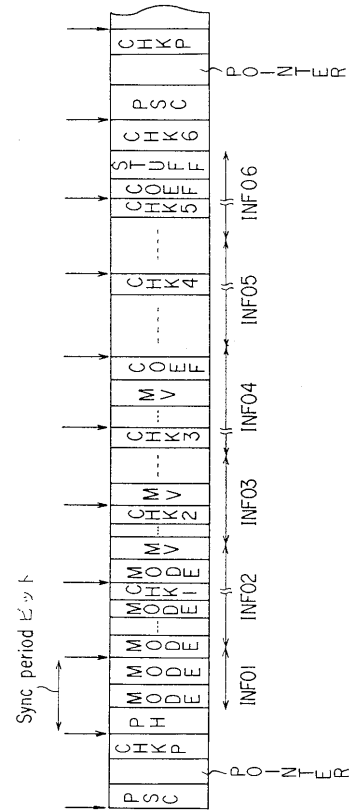
【 図 1 3 】



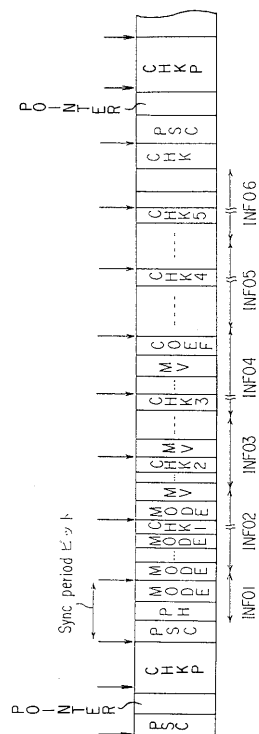
【図 14】



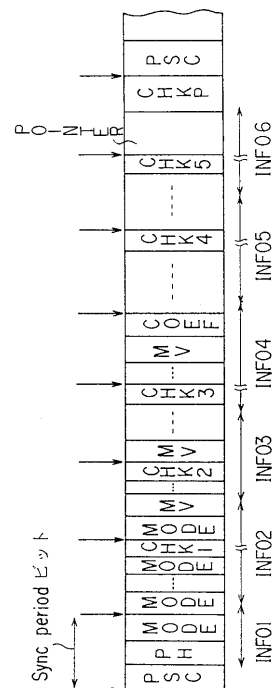
【図 15】



【図 16】

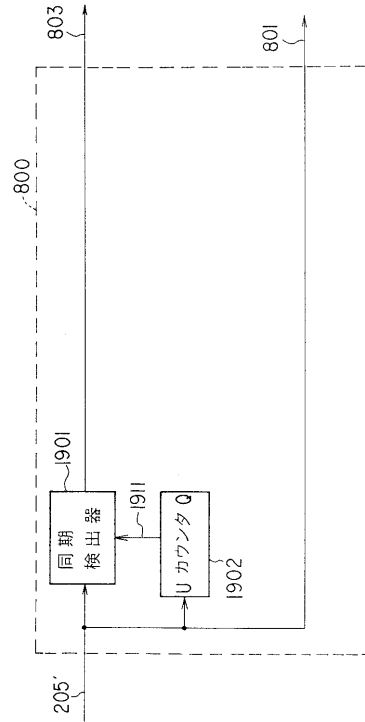


【図 17】

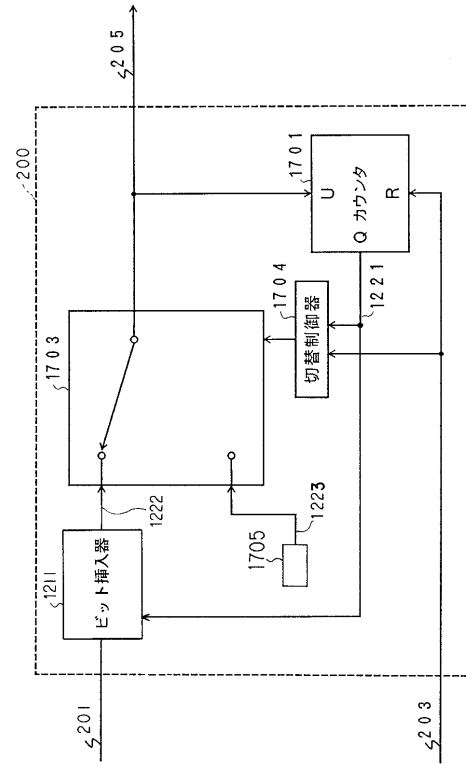




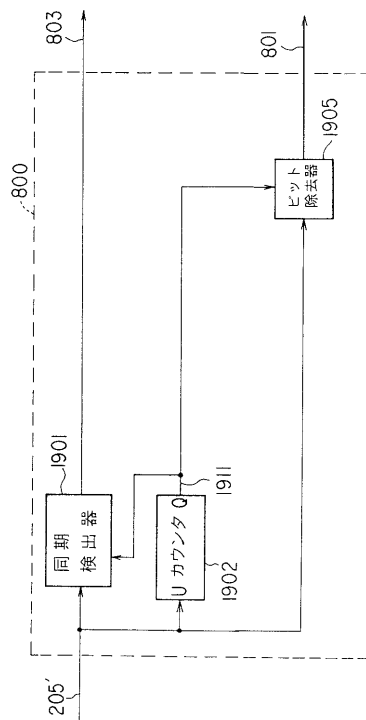
【 図 2 2 】



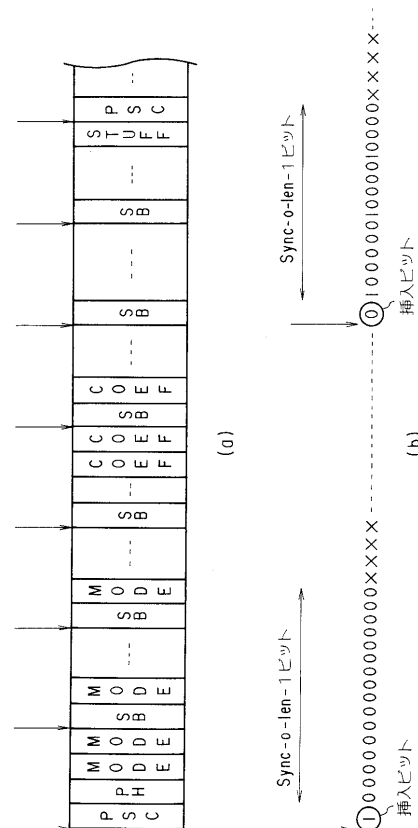
【 図 2 3 】



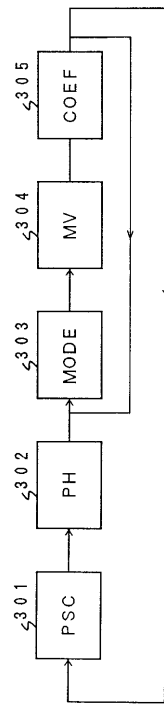
【 図 2 4 】



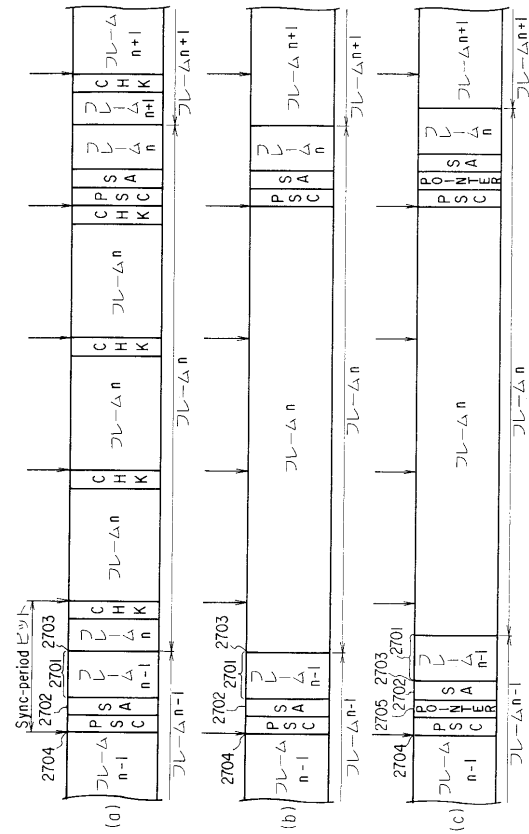
【 図 2 5 】



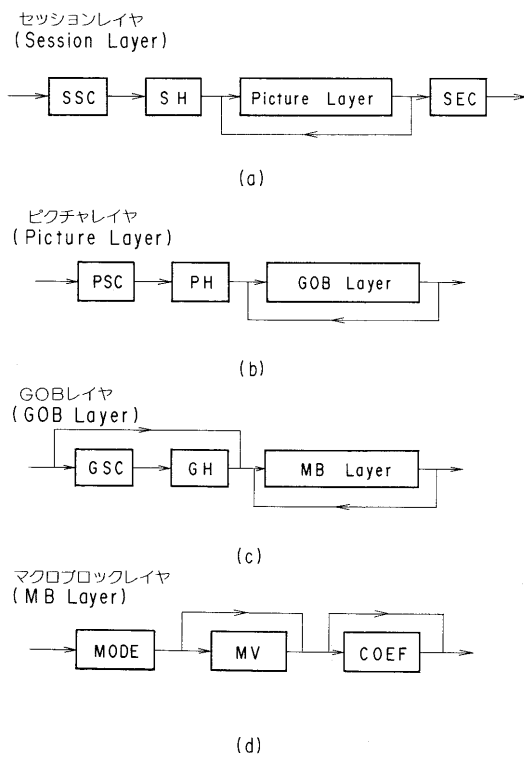
【図 26】



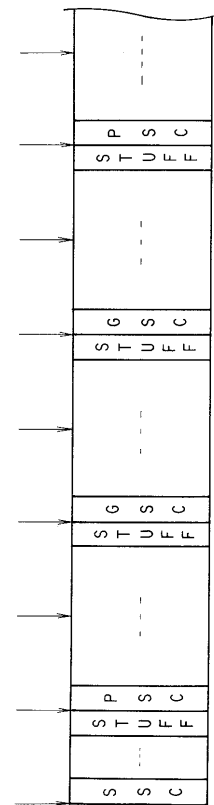
【図 27】



【図 28】



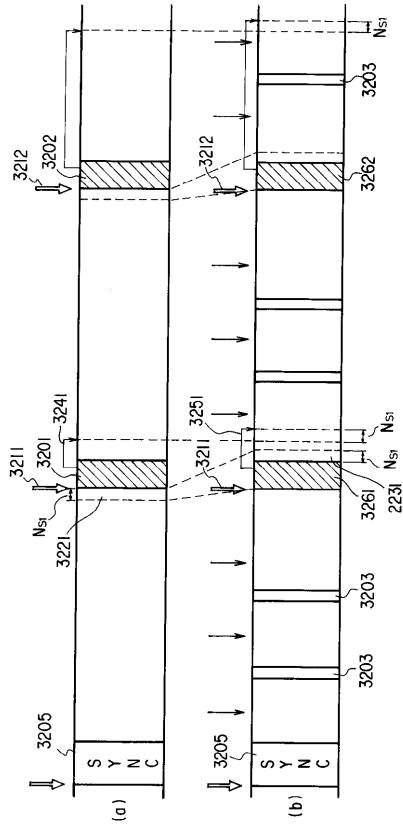
【図 29】







【図 35】



## フロントページの続き

(31)優先権主張番号 特願平8-232362

(32)優先日 平成8年9月2日(1996.9.2)

(33)優先権主張国 日本国(JP)

(74)代理人 100070437

弁理士 河井 将次

(72)発明者 菊池 義浩

神奈川県川崎市幸区小向東芝町1番地 株式会社東芝研究開発センター内

(72)発明者 渡邊 敏明

神奈川県川崎市幸区小向東芝町1番地 株式会社東芝研究開発センター内

(72)発明者 中條 健

神奈川県川崎市幸区小向東芝町1番地 株式会社東芝研究開発センター内

(72)発明者 永井 剛

神奈川県川崎市幸区小向東芝町1番地 株式会社東芝研究開発センター内

(72)発明者 駄竹 健志

東京都港区芝浦一丁目1番1号 株式会社東芝本社事務所内

審査官 矢頭 尚之

(56)参考文献 特開平07-235879(JP,A)

特開平07-038857(JP,A)

渡邊 菊池 他, MPEG4対応低ビットレート動画像符号化方式, 電子情報通信学会技術研究報告, 日本, 社団法人 電子情報通信学会, 1996年 3月, IE95-145, 37~44 頁

(58)調査した分野(Int.Cl.<sup>7</sup>, DB名)

H04L 1/00

H04L 7/00

H04N 7/24

H03M 7/30