

(19) 日本国特許庁 (JP)

(12) 特 許 公 報 (B2)

(11) 特許番号

特許第4482811号
(P4482811)

(45) 発行日 平成22年6月16日 (2010. 6. 16)

(24) 登録日 平成22年4月2日 (2010. 4. 2)

(51) Int. Cl.	F I
HO 4 N 7/32 (2006. 01)	HO 4 N 7/137 Z
HO 3 M 7/36 (2006. 01)	HO 3 M 7/36
HO 4 N 5/92 (2006. 01)	HO 4 N 5/92 H

請求項の数 16 (全 88 頁)

(21) 出願番号	特願2005-64630 (P2005-64630)	(73) 特許権者	000002185
(22) 出願日	平成17年3月8日 (2005. 3. 8)		ソニー株式会社
(62) 分割の表示	特願2000-599213 (P2000-599213)		東京都港区港南1丁目7番1号
	の分割	(74) 代理人	100082740
原出願日	平成12年2月9日 (2000. 2. 9)		弁理士 田辺 恵基
(65) 公開番号	特開2005-245002 (P2005-245002A)	(72) 発明者	北村 卓也
(43) 公開日	平成17年9月8日 (2005. 9. 8)		東京都品川区北品川6丁目7番35号ソニー株式会社内
審査請求日	平成18年4月14日 (2006. 4. 14)		
(31) 優先権主張番号	特願平11-31944	審査官	古川 哲也
(32) 優先日	平成11年2月9日 (1999. 2. 9)		
(33) 優先権主張国	日本国 (JP)		

最終頁に続く

(54) 【発明の名称】 記録装置及び方法

(57) 【特許請求の範囲】

【請求項 1】

過去 2 回以上の符号化処理または復号処理が施された符号化ストリームを再符号化ストリームに変換処理して記録媒体に記録する記録装置において、

前記符号化ストリームに対する過去の符号化処理または復号処理において利用された履歴符号化パラメータを、前記符号化ストリームとともに入力する入力手段と、

前記入力手段により入力された前記符号化ストリームを前記再符号化ストリームに変換処理する変換手段と、

前記履歴符号化パラメータの選択的な組み合わせを示す組み合わせ情報を生成する組み合わせ情報生成手段と、

前記組み合わせ情報生成手段により生成された前記組み合わせ情報及び前記組み合わせ情報の組み合わせに対応する前記履歴符号化パラメータを、前記再符号化ストリームとともに前記記録媒体に記録する記録手段と

を備え、

前記記録手段は、前記再符号化ストリームと前記履歴符号化パラメータとを、前記記録媒体の異なる領域に記録する

記録装置。

【請求項 2】

前記記録媒体は磁気テープであり、

前記記録手段は、前記再符号化ストリームをシンクコードに近い位置に記録し、前記履

歴符号化パラメータを前記シンクコードから前記再符号化ストリームの記録位置より離れた位置に記録する

請求項 1 に記載の記録装置。

【請求項 3】

現在の符号化パラメータを算出する符号化パラメータ算出手段を更に備え、

前記組み合わせ情報生成手段は、前記履歴符号化パラメータ及び現在の符号化パラメータの選択的な組み合わせを示す組み合わせ情報を生成する

請求項 1 に記載の記録装置。

【請求項 4】

前記変換手段は、前記入力手段により入力された前記履歴符号化パラメータの中から、変換処理において再利用する履歴符号化パラメータを再利用符号化パラメータとして選択し、選択した前記再利用符号化パラメータを再利用して前記再符号化ストリームに変換処理する

請求項 1 に記載の記録装置。

【請求項 5】

現在の符号化パラメータを算出する符号化パラメータ算出手段を更に備え、

前記変換手段は、前記入力手段により入力された前記履歴符号化パラメータ及び前記現在の符号化パラメータの中から、変換処理において再利用する履歴符号化パラメータを再利用符号化パラメータとして選択し、選択した前記再利用符号化パラメータを再利用して前記再符号化ストリームに変換処理する

請求項 1 に記載の記録装置。

【請求項 6】

前記履歴符号化パラメータ及び前記現在の符号化パラメータから前記再利用符号化パラメータを除いた符号化パラメータを含む符号化履歴情報を生成する生成手段を更に備え、

前記記録手段は、前記符号化履歴情報又は前記再利用符号化パラメータを、前記再符号化ストリームとともに前記記録媒体に記録する

請求項 5 に記載の記録装置。

【請求項 7】

前記組み合わせ情報は、変換処理による画質劣化の程度に応じて区別されている情報である

請求項 1 に記載の記録装置。

【請求項 8】

前記組み合わせ情報は、前記再符号化ストリームを利用するアプリケーションに応じて区別されている情報であり、

前記組み合わせ情報生成手段は、前記アプリケーションからの要求に基づいて、上記組み合わせ情報を生成する

請求項 1 に記載の記録装置。

【請求項 9】

前記組み合わせ情報は、前記再符号化ストリームが伝送される伝送路、又は、前記記録媒体の容量に応じて区別されている情報である

請求項 1 に記載の記録装置。

【請求項 10】

前記組み合わせ情報は、前記履歴符号化パラメータの全てを出力するか、前記履歴符号化パラメータのうちの一部を出力するかを識別する第 1 の識別情報と、前記履歴符号化パラメータのうちの一部を出力する場合の組み合わせを識別する第 2 の識別情報と

を含む請求項 1 に記載の記録装置。

【請求項 11】

前記組み合わせ情報は、前記履歴符号化パラメータ及び前記現在の符号化パラメータの全てを出力するか、前記履歴符号化パラメータ及び前記現在の符号化パラメータのうちの一部を出力するかを識別する第 1 の識別情報と、前記履歴符号化パラメータ及び前記現在

10

20

30

40

50

の符号化パラメータのうちの一部を出力する場合の組み合わせを識別する第2の識別情報と

を含む請求項5に記載の記録装置。

【請求項12】

前記変換手段は、ビットレート又はGOP構造を変更するように変換処理を行う

請求項1に記載の記録装置。

【請求項13】

前記変換手段は、シーケンスレイヤ、GOPレイヤ、ピクチャレイヤ、スライスレイヤ、および、マクロブロックレイヤを有するMPEG方式で変換処理を行う

請求項1に記載の記録装置。

10

【請求項14】

前記履歴符号化パラメータは、前記符号化ストリームに対する過去複数世代の符号化パラメータである

請求項1に記載の記録装置。

【請求項15】

前記履歴符号化パラメータは、前記符号化ストリームに対する直近過去4世代分の符号化パラメータである

請求項14に記載の記録装置。

【請求項16】

過去2回以上の符号化処理または復号処理が施された符号化ストリームを再符号化ストリームに変換処理して記録媒体に記録する記録方法において、

20

前記符号化ストリームに対する過去の符号化処理または復号処理において利用された履歴符号化パラメータを、前記符号化ストリームとともに入力する入力ステップと、

前記入力手段により入力された前記符号化ストリームを前記再符号化ストリームに変換処理する変換ステップと、

前記履歴符号化パラメータの選択的な組み合わせを示す組み合わせ情報を生成する組み合わせ情報生成ステップと、

前記組み合わせ情報生成手段により生成された前記組み合わせ情報及び前記組み合わせ情報の組み合わせに対応する前記履歴符号化パラメータを、前記再符号化ストリームとともに前記記録媒体に記録する記録ステップと

30

を有し、

前記記録ステップでは、前記再符号化ストリームと前記履歴符号化パラメータとを、前記記録媒体の異なる領域に記録する

記録方法。

【発明の詳細な説明】

【技術分野】

【0001】

本発明は、コーディングシステム及び方法、符号化装置および方法、復号化装置及び方法、記録装置及び方法、ならびに再生装置及び方法に関し、特に、MPEG規格に基づいて過去に符号化処理が施されたことがある符号化ストリームに対して再符号化処理を施すことによって、異なるGOP (Group of Pictures) 構造や異なるビットレートを有した再符号化ストリームを生成するためのトランスコードに用いて好適なコーディングシステム及び方法、符号化装置および方法、復号化装置及び方法、記録装置及び方法、ならびに再生装置及び方法に関する。

40

【背景技術】

【0002】

近年、テレビジョンプログラムを制作及び放送する放送局においては、ビデオデータを圧縮/符号化処理するために、MPEG (Moving Picture Experts Group) 技術が一般的に使われるようになってきた。特に、ビデオデータをテープなどのランダムアクセス可能な記

50

録媒体素材に記録する場合、及びビデオデータをケーブルや衛星を介して伝送する場合には、このMPEG技術がデファクトスタンダードになりつつある。

【0003】

放送局において制作されたビデオプログラムが各家庭に伝送されるまでの放送局における処理の一例を簡単に説明する。まず、ビデオカメラとVTR (Video Tape Recorder) が一体となったカムコーダに設けられたエンコーダによって、ソースビデオデータをエンコード処理して磁気テープ上に記録する。この際、カムコーダのエンコーダは、VTRのテープの記録フォーマットに適するように、ソースビデオデータを符号化する。たとえば、この磁気テープ上に記録されるMPEGビットストリームのGOP構造は、2フレームから1GOPが構成される構造（たとえば、I, B, I, B, I, B, ……）とされる。また磁気テープ上に記録されているMPEGビットストリームのビットレートは、18Mbpsである。

10

【0004】

次に、メイン放送局において、この磁気テープ上に記録されたビデオビットストリームを編集する編集処理を行う。そのために、磁気テープ上に記録されたビデオビットストリームのGOP構造を、編集処理に適したGOP構造に変換する。編集処理に適したGOP構造とは、1GOPが1フレームから構成され、すべてのピクチャがIピクチャであるGOP構造である。なぜなら、フレーム単位で編集を行うためには、他のピクチャと相関のないIピクチャがもっとも適しているからである。実際のオペレーションとしては、磁気テープ上に記録されたビデオストリームを一旦デコードしてベースバンドのビデオデータに戻す。そして、そのベースバンドのビデオ信号を、すべてのピクチャがIピクチャとなるように再エンコードする。このようにデコード処理及び再エンコード処理を行うことによって、編集処理に適したGOP構造を有したビットストリームを生成することができる。

20

【0005】

次に、上述した編集処理によって生成された編集ビデオプログラムを、メイン局から地方局に伝送するために、編集ビデオプログラムのビットストリームを、伝送処理に適したGOP構造及びビットレートに変換する。放送局間の伝送に適したGOP構造とは、たとえば、1GOPが15フレームから構成されているGOP構造（たとえば、I, B, B, P, B, B, P ……）である。また、放送局間の伝送に適したビットレートは、一般的に放送局間においては、光ファイバなどの高伝送容量を有した専用線が設けられているので、50Mbps以上のハイビットレートであることが望ましい。具体的には、編集処理されたビデオプログラムのビットストリームを一旦デコードしてベースバンドのビデオデータに戻す。そして、そのベースバンドのビデオデータを上述した放送局間の伝送に適したGOP構造及びビットレートを有するように再エンコードする。

30

【0006】

地方局においては、メイン局から伝送されてきたビデオプログラムの中に、地方特有のコマーシャルを挿入するために編集処理が行われる。つまり、上述した編集処理と同じように、メイン局から伝送されてきたビデオストリームを一旦デコードしてベースバンドのビデオデータに戻す。そして、そのベースバンドのビデオ信号を、すべてのピクチャがIピクチャとなるように再エンコードすることによって、編集処理に適したGOP構造を有したビットストリームを生成することができる。

40

【0007】

続いて、この地方局において編集処理が行われたビデオプログラムを各家庭に、ケーブルや衛星を介して伝送するために、この伝送処理に適したGOP構造及びビットレートに変換する。たとえば、各家庭に伝送するための伝送処理に適したGOP構造とは、1GOPが15フレームから構成されるGOP構造（たとえば、I, B, B, P, B, B, P ……）であって、各家庭に伝送するための伝送処理に適したビットレートは、5Mbps程度の低ビットレートである。具体的には、編集処理されたビデオプログラムのビットストリームを一旦デコードしてベースバンドのビデオデータに戻す。そして、そのベースバンドのビデオデータを上述した伝送処理に適したGOP構造及びビットレートを有するように再エンコードする。

50

【0008】

以上の説明からも理解できるように、放送局から各家庭にビデオプログラムが伝送される間に、複数回の復号処理及び符号化処理が繰り返されている。実際には、放送局における処理は上述した信号処理以外にもさまざまな信号処理が必要であり、そのたびに復号処理及び符号化処理を繰り返さなければならない。

【発明の開示】

【発明が解決しようとする課題】

【0009】

しかしながら、MPEG規格に基づく符号化処理及び復号処理は、100%可逆の処理ではないことは良く知られている。つまり、エンコードされる前のベースバンドのビデオデータと、デコードされた後のビデオデータは100%同じでは無く、この符号化処理及び復号処理によって画質が劣化している。つまり、上述したように、デコード処理及びエンコード処理を繰り返すと、その処理の度に、画質が劣化してしまうという問題があった。別の言葉で表現すると、デコード/エンコード処理を繰り返す毎に、画質の劣化が蓄積されてしまう。

【0010】

本発明は、このような状況に鑑みてなされたものであり、MPEG規格に基づいて符号化された符号化ビットストリームのGOP (Group of Pictures) の構造を変更するために復号及び符号化処理を繰り返したとしても画質劣化の発生しないトランスコーディングシステムを実現できるようにするものである。

【課題を解決するための手段】

【0011】

本発明は、過去2回以上の符号化処理または復号処理が施された符号化ストリームを再符号化ストリームに変換処理して記録媒体に記録する記録装置において、符号化ストリームに対する過去の符号化処理または復号処理において利用された履歴符号化パラメータを、符号化ストリームとともに入力する入力手段と、入力手段により入力された符号化ストリームを再符号化ストリームに変換処理する変換手段と、履歴符号化パラメータの選択的な組み合わせを示す組み合わせ情報を生成する組み合わせ情報生成手段と、組み合わせ情報生成手段により生成された組み合わせ情報及び組み合わせ情報の組み合わせに対応する履歴符号化パラメータを、再符号化ストリームとともに記録媒体に記録する記録手段とを備え、記録手段は、再符号化ストリームと履歴符号化パラメータとを、記録媒体の異なる領域に記録するようにした。

【0012】

また本発明は、過去2回以上の符号化処理または復号処理が施された符号化ストリームを再符号化ストリームに変換処理して記録媒体に記録する記録方法において、符号化ストリームに対する過去の符号化処理または復号処理において利用された履歴符号化パラメータを、符号化ストリームとともに入力する入力ステップと、入力手段により入力された符号化ストリームを再符号化ストリームに変換処理する変換ステップと、履歴符号化パラメータの選択的な組み合わせを示す組み合わせ情報を生成する組み合わせ情報生成ステップと、組み合わせ情報生成手段により生成された組み合わせ情報及び組み合わせ情報の組み合わせに対応する履歴符号化パラメータを、再符号化ストリームとともに記録媒体に記録する記録ステップとを有し、記録ステップでは、再符号化ストリームと履歴符号化パラメータとを、記録媒体の異なる領域に記録するようにした。

【発明の効果】

【0014】

本発明のトランスコーダによれば、後段のアプリケーションに適応した最小限の符号化パラメータで、ビデオデータの再符号化処理を繰り返したとしても、画質劣化を最小限に留めることができるコーディングシステムを実現することができる。

【0015】

本発明のトランスコーダは、ソース符号化ストリームを復号してビデオデータを生成するとともに、ソース符号化ストリームから過去の符号化処理により生成された過去の符号化パラメータを抽出する復号手段と、ビデオデータを再符号化し、再符号化ビデオストリームを生成する符号化手段と、過去の符号化パラメータを受け取り、過去の符号化パラメータに基づいて、符号化手段の再符号化処理を制御するとともに、過去の符号化パラメータを再符号化ストリーム中に選択的に記述する制御手段とを備えている。

【0016】

本発明のトランスコーダにおける符号化装置は、過去の符号化パラメータから、符号化手段の後段に接続されるアプリケーションにおいて必要となる符号化パラメータを選択し、該選択された過去の符号化パラメータを、再符号化ストリーム中に記述する。

10

【0017】

本発明のトランスコーダにおける符号化装置は、過去の符号化パラメータを選択的に再符号化ストリーム中に記述するとともに、再符号化ストリーム中に記述される過去の符号化パラメータのデータセットを示すフラグ又は／及びインジケータを、再符号化ストリーム中に記述する。

【0018】

本発明のトランスコーダにおける符号化装置は、過去の符号化パラメータに関する情報を、符号化ストリーム中にhistory_stream()として記述するとともに、再符号化ストリームに関する情報を、re_coding_stream_info()として再符号化ストリーム中に記述する。

20

【0019】

本発明のトランスコーダにおける符号化装置は、過去の符号化パラメータを選択的に再符号化ストリーム中にhistory_stream()として記述するとともに、再符号化ストリーム中に記述される過去の符号化パラメータのデータセットに関する情報を、re_coding_stream_info()として再符号化ストリーム中に記述する。

【0020】

本発明の符号化装置は、ビデオデータに関して施された過去の符号化処理に関する過去の符号化パラメータを受け取り、過去の符号化パラメータを選択的に再符号化ストリーム中に記述するとともに、再符号化ストリーム中に記述される過去の符号化パラメータのデータセットを示す情報を、再符号化ストリーム中に記述する。

30

【0021】

本発明の復号化装置は、符号化ストリームから、符号化ストリーム中に重畳されている過去の符号化パラメータのデータセットに関する情報を抽出し、このデータセットに関する情報に基づいて、符号化ストリームから、過去の符号化パラメータを抽出する。

【0022】

本発明の復号化装置は、符号化ストリームから、符号化ストリーム中にre_coding_stream_info()として記述されているフラグ又は／及びインジケータを抽出し、このフラグ又は／及びインジケータに基づいて、符号化ストリームから、過去の符号化パラメータを抽出する。

【発明を実施するための最良の形態】

40

【0023】

以下に、本発明を適用したトランスコーダについて説明するが、その前に、動画像信号の圧縮符号化について説明する。なお、本明細書においてシステムの用語は、複数の装置、手段などにより構成される全体的な装置を意味するものである。

【0024】

例えば、テレビ会議システム、テレビ電話システムなどのように、動画像信号を遠隔地に伝送するシステムにおいては、伝送路を効率良く利用するため、映像信号のライン相関やフレーム間相関を利用して、画像信号を圧縮符号化するようになされている。

【0025】

ライン相関を利用すると、画像信号を、例えばDCT（離散コサイン変換）処理するなど

50

して圧縮することができる。

【0026】

また、フレーム間相関を利用すると、画像信号をさらに圧縮して符号化することが可能となる。例えば図1に示すように、時刻 t_1 乃至 t_3 において、フレーム画像PC1乃至PC3がそれぞれ発生している場合、フレーム画像PC1およびPC2の画像信号の差を演算して、PC12を生成し、また、フレーム画像PC2およびPC3の差を演算して、PC23を生成する。通常、時間的に隣接するフレームの画像は、それ程大きな変化を有していないため、両者の差を演算すると、その差分信号は小さな値のものとなる。そこで、この差分信号を符号化すれば、符号量を圧縮することができる。

【0027】

しかしながら、差分信号のみを伝送したのでは、元の画像を復元することができない。そこで、各フレームの画像を、Iピクチャ、PピクチャまたはBピクチャの3種類のピクチャタイプのいずれかとし、画像信号を圧縮符号化するようにしている。

【0028】

すなわち、例えば図2に示すように、フレームF1乃至F17までの17フレームの画像信号をグループオブピクチャ(GOP)とし、処理の1単位とする。そして、その先頭のフレームF1の画像信号はIピクチャとして符号化し、第2番目のフレームF2はBピクチャとして、また第3番目のフレームF3はPピクチャとして、それぞれ処理する。以下、第4番目以降のフレームF4乃至F17は、BピクチャまたはPピクチャとして交互に処理する。

【0029】

Iピクチャの画像信号としては、その1フレーム分の画像信号をそのまま伝送する。これに対して、Pピクチャの画像信号としては、基本的には、図2に示すように、それより時間的に先行するIピクチャまたはPピクチャの画像信号からの差分を伝送する。さらにBピクチャの画像信号としては、基本的には、図3に示すように、時間的に先行するフレームまたは後行するフレームの両方の平均値からの差分を求め、その差分を符号化する。

【0030】

図4は、このようにして、動画像信号を符号化する方法の原理を示している。同図に示すように、最初のフレームF1は、Iピクチャとして処理されるため、そのまま伝送データF1Xとして伝送路に伝送される(画像内符号化)。これに対して、第2のフレームF2は、Bピクチャとして処理されるため、時間的に先行するフレームF1と、時間的に後行するフレームF3の平均値との差分が演算され、その差分が伝送データF2Xとして伝送される。

【0031】

ただし、このBピクチャとしての処理は、さらに細かく説明すると、4種類存在する。その第1の処理は、元のフレームF2のデータをそのまま伝送データF2Xとして伝送するものであり(SP1)(イントラ符号化)、Iピクチャにおける場合と同様の処理となる。第2の処理は、時間的に後のフレームF3からの差分を演算し、その差分(SP2)を伝送するものである(後方予測符号化)。第3の処理は、時間的に先行するフレームF1との差分(SP3)を伝送するものである(前方予測符号化)。さらに第4の処理は、時間的に先行するフレームF1と後行するフレームF3の平均値との差分(SP4)を生成し、これを伝送データF2Xとして伝送するものである(両方向予測符号化)。

【0032】

実際には、上述した4つの方法のうちの伝送データが最も少なくなる方法が採用される。

【0033】

なお、差分データを伝送するとき、差分を演算する対象となるフレームの画像(参照画像)との間の動きベクトル x_1 (フレームF1とF2の間の動きベクトル)(前方予測の場合)、もしくは x_2 (フレームF3とF2の間の動きベクトル)(後方予測の場合)、または x_1 と x_2 の両方(両方向予測の場合)が、差分データとともに伝送される。

【0034】

また、PピクチャのフレームF3は、時間的に先行するフレームF1を参照画像として、このフレームとの差分信号(SP3)と、動きベクトルx3が演算され、これが伝送データF3Xとして伝送される(前方予測符号化)。あるいはまた、元のフレームF3のデータが、そのままデータF3Xとして伝送される(SP1)(イントラ符号化)。これらの方法のうち、Bピクチャにおける場合と同様に、伝送データがより少なくなる方法が選択される。

【0035】

図5は、上述した原理に基づいて、動画像信号を符号化して伝送し、これを復号する装置の構成例を示している。符号化装置1は、入力された映像信号を符号化し、伝送路としての記録媒体3に伝送するようになされている。そして、復号装置2は、記録媒体3に記録された信号を再生し、これを復号して出力するようになされている。

10

【0036】

符号化装置1においては、入力された映像信号が前処理回路11に入力され、そこで輝度信号と色信号(本実施の形態の場合、色差信号)が分離され、それぞれA/D変換器12, 13でアナログ信号からデジタル信号に変換される。A/D変換器12, 13によりデジタル信号に変換された映像信号は、フレームメモリ14に供給され、記憶される。フレームメモリ14は、輝度信号を輝度信号フレームメモリ15に、また、色差信号を色差信号フレームメモリ16に、それぞれ記憶させる。

20

【0037】

フォーマット変換回路17は、フレームメモリ14に記憶されたフレームフォーマットの信号を、ブロックフォーマットの信号に変換する。すなわち、図6に示すように、フレームメモリ14に記憶された映像信号は、1ライン当りHドットのラインがVライン集められた、図6(A)に示すようなフレームフォーマットのデータとされている。フォーマット変換回路17は、この1フレームの信号を、図6(B)に示すように、16ラインを単位としてM個のスライスに区分する。そして、各スライスは、M個のマクロブロックに分割される。マクロブロックは、図6(C)に示すように、16×16個の画素(ドット)に対応する輝度信号により構成され、この輝度信号は、さらに8×8ドットを単位とするブロックY[1]乃至Y[4]に区分される。そして、この16×16ドットの輝度信号には、8×8ドットのCb信号と、8×8ドットのCr信号が対応される。

30

【0038】

このように、ブロックフォーマットに変換されたデータは、フォーマット変換回路17からエンコーダ18に供給され、ここでエンコード(符号化)が行われる。その詳細については、図7を参照して後述する。

【0039】

エンコーダ18によりエンコードされた信号は、ビットストリームとして伝送路に出力される。例えば記録回路19に供給され、デジタル信号として記録媒体3に記録される。

復号装置2の再生回路30により記録媒体3より再生されたデータは、デコーダ31に供給され、デコードされる。デコーダ31の詳細については、図12を参照して後述する。

40

【0040】

デコーダ31によりデコードされたデータは、フォーマット変換回路32に入力され、ブロックフォーマットからフレームフォーマットに変換される。そして、フレームフォーマットの輝度信号は、フレームメモリ33の輝度信号フレームメモリ34に供給されて記憶され、色差信号は色差信号フレームメモリ35に供給されて記憶される。輝度信号フレームメモリ34と色差信号フレームメモリ35から読み出された輝度信号と色差信号は、それぞれD/A変換器36, 37によりアナログ信号に変換され、後処理回路38に供給される。後処理回路38は、輝度信号と色差信号を合成して出力する。

【0041】

次に図7を参照して、エンコーダ18の構成について説明する。符号化される画像デー

50

タは、マクロブロック単位で動きベクトル検出回路50に入力される。動きベクトル検出回路50は、予め設定されている所定のシーケンスに従って、各フレームの画像データを、Iピクチャ、Pピクチャ、またはBピクチャとして処理する。シーケンシャルに入力される各フレームの画像を、I、P、またはBのいずれのピクチャとして処理するかは、予め定められている（例えば、図2と図3に示したように、フレームF1乃至F17により構成されるグループオブピクチャが、I、B、P、B、P、・・・B、Pとして処理される）。

【0042】

Iピクチャとして処理されるフレーム（例えば、フレームF1）の画像データは、動きベクトル検出回路50からフレームメモリ51の前方原画像部51aに転送、記憶され、Bピクチャとして処理されるフレーム（例えば、フレームF2）の画像データは、原画像部51bに転送、記憶され、Pピクチャとして処理されるフレーム（例えば、フレームF3）の画像データは、後方原画像部51cに転送、記憶される。

【0043】

また、次のタイミングにおいて、さらにBピクチャ（フレームF4）またはPピクチャ（フレームF5）として処理すべきフレームの画像が入力されたとき、それまで後方原画像部51cに記憶されていた最初のPピクチャ（フレームF3）の画像データが、前方原画像部51aに転送され、次のBピクチャ（フレームF4）の画像データが、参照原画像部51bに記憶（上書き）され、次のPピクチャ（フレームF5）の画像データが、後方原画像部51cに記憶（上書き）される。このような動作が順次繰り返される。

【0044】

フレームメモリ51に記憶された各ピクチャの信号は、そこから読み出され、予測モード切り替え回路52において、フレーム予測モード処理、またはフィールド予測モード処理が行なわれる。

【0045】

さらにまた、予測判定回路54の制御の下に、演算器53において、画像内予測、前方予測、後方予測、または両方向予測の演算が行なわれる。これらの処理のうち、いずれの処理を行なうかは、予測誤差信号（処理の対象とされている参照画像と、これに対する予測画像との差分）に対応して決定される。このため、動きベクトル検出回路50は、この判定に用いられる予測誤差信号の絶対値和（自乗和でもよい）を生成する。

【0046】

ここで、予測モード切り替え回路52におけるフレーム予測モードとフィールド予測モードについて説明する。

【0047】

フレーム予測モードが設定された場合においては、予測モード切り替え回路52は、動きベクトル検出回路50より供給される4個の輝度ブロックY[1]乃至Y[4]を、そのまま後段の演算器53に出力する。すなわち、この場合においては、図8に示すように、各輝度ブロックに奇数フィールドのラインのデータと、偶数フィールドのラインのデータとが混在した状態となっている。このフレーム予測モードにおいては、4個の輝度ブロック（マクロブロック）を単位として予測が行われ、4個の輝度ブロックに対して1個の動きベクトルが対応される。

【0048】

これに対して、予測モード切り替え回路52は、フィールド予測モードにおいては、図8に示す構成で動きベクトル検出回路50より入力される信号を、図9に示すように、4個の輝度ブロックのうち、輝度ブロックY[1]とY[2]を、例えば奇数フィールドのラインのドットだけで構成させ、他の2個の輝度ブロックY[3]とY[4]を、偶数フィールドのラインのドットだけで構成させて、演算器53に出力する。この場合においては、2個の輝度ブロックY[1]とY[2]に対して、1個の動きベクトルが対応され、他の2個の輝度ブロックY[3]とY[4]に対して、他の1個の動きベクトルが対応される。

【0049】

動きベクトル検出回路50は、フレーム予測モードにおける予測誤差の絶対値和、およびフィールド予測モードにおける予測誤差の絶対値和を予測モード切り替え回路52に出力する。予測モード切り替え回路52は、フレーム予測モードとフィールド予測モードにおける予測誤差の絶対値和を比較し、その値が小さい予測モードに対応する処理を施して、データを演算器53に出力する。

【0050】

ただし、このような処理は、実際には動きベクトル検出回路50で行われる。すなわち、動きベクトル検出回路50は、決定されたモードに対応する構成の信号を予測モード切り替え回路52に出力し、予測モード切り替え回路52は、その信号を、そのまま後段の演算器53に出力する。

10

【0051】

なお、色差信号は、フレーム予測モードの場合、図8に示すように、奇数フィールドのラインのデータと偶数フィールドのラインのデータとが混在する状態で、演算器53に供給される。また、フィールド予測モードの場合、図9に示すように、各色差ブロックCb, Crの上半分(4ライン)が、輝度ブロックY[1], Y[2]に対応する奇数フィールドの色差信号とされ、下半分(4ライン)が、輝度ブロックY[3], Y[4]に対応する偶数フィールドの色差信号とされる。

【0052】

また、動きベクトル検出回路50は、以下に示すようにして、予測判定回路54において、画像内予測、前方予測、後方予測、または両方向予測のいずれの予測を行なうかを決定するための予測誤差の絶対値和を生成する。

20

【0053】

すなわち、画像内予測の予測誤差の絶対値和として、参照画像のマクロブロックの信号 A_{ij} の総和 ΣA_{ij} の絶対値 $|\Sigma A_{ij}|$ と、マクロブロックの信号 A_{ij} の絶対値 $|A_{ij}|$ の総和 $\Sigma |A_{ij}|$ の差を求める。また、前方予測の予測誤差の絶対値和として、参照画像のマクロブロックの信号 A_{ij} と、予測画像のマクロブロックの信号 B_{ij} の差 $A_{ij} - B_{ij}$ の絶対値 $|A_{ij} - B_{ij}|$ の総和 $\Sigma |A_{ij} - B_{ij}|$ を求める。また、後方予測と両方向予測の予測誤差の絶対値和も、前方予測における場合と同様に(その予測画像を前方予測における場合と異なる予測画像に変更して)求める。

30

【0054】

これらの絶対値和は、予測判定回路54に供給される。予測判定回路54は、前方予測、後方予測および両方向予測の予測誤差の絶対値和のうちの最も小さいものを、インタ予測の予測誤差の絶対値和として選択する。さらに、このインタ予測の予測誤差の絶対値和と、画像内予測の予測誤差の絶対値和とを比較し、その小さい方を選択し、この選択した絶対値和に対応するモードを予測モードとして選択する。すなわち、画像内予測の予測誤差の絶対値和の方が小さければ、画像内予測モードが設定される。インタ予測の予測誤差の絶対値和の方が小さければ、前方予測、後方予測または両方向予測モードのうちの対応する絶対値和が最も小さかったモードが設定される。

【0055】

40

このように、動きベクトル検出回路50は、参照画像のマクロブロックの信号を、フレームまたはフィールド予測モードのうち、予測モード切り替え回路52により選択されたモードに対応する構成で、予測モード切り替え回路52を介して演算器53に供給するとともに、4つの予測モードのうちの予測判定回路54により選択された予測モードに対応する予測画像と参照画像の間の動きベクトルを検出し、可変長符号化回路58と動き補償回路64に出力する。上述したように、この動きベクトルとしては、対応する予測誤差の絶対値和が最小となるものが選択される。

【0056】

予測判定回路54は、動きベクトル検出回路50が前方原画像部51aよりIピクチャの画像データを読み出しているとき、予測モードとして、フレームまたはフィールド(画

50

像)内予測モード(動き補償を行わないモード)を設定し、演算器53のスイッチ53dを接点a側に切り替える。これにより、Iピクチャの画像データがDCTモード切り替え回路55に入力される。

【0057】

DCTモード切り替え回路55は、図10または図11に示すように、4個の輝度ブロックのデータを、奇数フィールドのラインと偶数フィールドのラインが混在する状態(フレームDCTモード)、または、分離された状態(フィールドDCTモード)、のいずれかの状態にして、DCT回路56に出力する。

【0058】

すなわち、DCTモード切り替え回路55は、奇数フィールドと偶数フィールドのデータを混在してDCT処理した場合における符号化効率と、分離した状態においてDCT処理した場合の符号化効率とを比較し、符号化効率の良好なモードを選択する。

10

【0059】

例えば、入力された信号を、図10に示すように、奇数フィールドと偶数フィールドのラインが混在する構成とし、上下に隣接する奇数フィールドのラインの信号と偶数フィールドのラインの信号の差を演算し、さらにその絶対値の和(または自乗和)を求める。

【0060】

また、入力された信号を、図11に示すように、奇数フィールドと偶数フィールドのラインが分離した構成とし、上下に隣接する奇数フィールドのライン同士の信号の差と、偶数フィールドのライン同士の信号の差を演算し、それぞれの絶対値の和(または自乗和)を求める。

20

【0061】

さらに、両者(絶対値和)を比較し、小さい値に対応するDCTモードを設定する。すなわち、前者の方が小さければ、フレームDCTモードを設定し、後者の方が小さければ、フィールドDCTモードを設定する。

【0062】

そして、選択したDCTモードに対応する構成のデータをDCT回路56に出力するとともに、選択したDCTモードを示すDCTフラグを、可変長符号化回路58、および動き補償回路64に出力する。

【0063】

30

予測モード切り替え回路52における予測モード(図8と図9)と、このDCTモード切り替え回路55におけるDCTモード(図10と図11)を比較して明らかなように、輝度ブロックに関しては、両者の各モードにおけるデータ構造は実質的に同一である。

【0064】

予測モード切り替え回路52において、フレーム予測モード(奇数ラインと偶数ラインが混在するモード)が選択された場合、DCTモード切り替え回路55においても、フレームDCTモード(奇数ラインと偶数ラインが混在するモード)が選択される可能性が高く、また予測モード切り替え回路52において、フィールド予測モード(奇数フィールドと偶数フィールドのデータが分離されたモード)が選択された場合、DCTモード切り替え回路55において、フィールドDCTモード(奇数フィールドと偶数フィールドのデータが分離されたモード)が選択される可能性が高い。

40

【0065】

しかしながら、必ずしも常にこのようにモードが選択されるわけではなく、予測モード切り替え回路52においては、予測誤差の絶対値和が小さくなるようにモードが決定され、DCTモード切り替え回路55においては、符号化効率が良好となるようにモードが決定される。

【0066】

DCTモード切り替え回路55より出力されたIピクチャの画像データは、DCT回路56に入力されてDCT処理され、DCT係数に変換される。このDCT係数は、量子化回路57に入力され、送信バッファ59のデータ蓄積量(バッファ蓄積量)に対応した量子化スケールで

50

量子化された後、可変長符号化回路 58 に入力される。

【0067】

可変長符号化回路 58 は、量子化回路 57 より供給される量子化スケール（スケール）に対応して、量子化回路 57 より供給される画像データ（いまの場合、I ピクチャのデータ）を、例えばハフマン符号などの可変長符号に変換し、送信バッファ 59 に出力する。

【0068】

可変長符号化回路 58 にはまた、量子化回路 57 より量子化スケール（スケール）、予測判定回路 54 より予測モード（画像内予測、前方予測、後方予測、または両方向予測のいずれが設定されたかを示すモード）、動きベクトル検出回路 50 より動きベクトル、予測モード切り替え回路 52 より予測フラグ（フレーム予測モードまたはフィールド予測モードのいずれが設定されたかを示すフラグ）、および DCT モード切り替え回路 55 が出力する DCT フラグ（フレーム DCT モードまたはフィールド DCT モードのいずれが設定されたかを示すフラグ）が入力されており、これらも可変長符号化される。

【0069】

送信バッファ 59 は、入力されたデータを一時蓄積し、蓄積量に対応するデータを量子化回路 57 に出力する。送信バッファ 59 は、そのデータ残量が許容上限値まで増量すると、量子化制御信号によって量子化回路 57 の量子化スケールを大きくすることにより、量子化データのデータ量を低下させる。また、これとは逆に、データ残量が許容下限値まで減少すると、送信バッファ 59 は、量子化制御信号によって量子化回路 57 の量子化スケールを小さくすることにより、量子化データのデータ量を増大させる。このようにして、送信バッファ 59 のオーバーフローまたはアンダフローが防止される。

【0070】

そして、送信バッファ 59 に蓄積されたデータは、所定のタイミングで読み出され、伝送路に出力され、例えば記録回路 19 を介して記録媒体 3 に記録される。

【0071】

一方、量子化回路 57 より出力された I ピクチャのデータは、逆量子化回路 60 に入力され、量子化回路 57 より供給される量子化スケールに対応して逆量子化される。逆量子化回路 60 の出力は、IDCT（逆離散コサイン変換）回路 61 に入力され、逆離散コサイン変換処理された後、演算器 62 を介してフレームメモリ 63 の前方予測画像部 63a 供給されて記憶される。

【0072】

動きベクトル検出回路 50 は、シーケンシャルに入力される各フレームの画像データを、たとえば、I、B、P、B、P、B・・・のピクチャとしてそれぞれ処理する場合、最初に入力されたフレームの画像データを I ピクチャとして処理した後、次に入力されたフレームの画像を B ピクチャとして処理する前に、さらにその次に入力されたフレームの画像データを P ピクチャとして処理する。B ピクチャは、後方予測を伴うため、後方予測画像としての P ピクチャが先に用意されていないと、復号することができないからである。

【0073】

そこで動きベクトル検出回路 50 は、I ピクチャの処理の次に、後方原画像部 51c に記憶されている P ピクチャの画像データの処理を開始する。そして、上述した場合と同様に、マクロブロック単位でのフレーム間差分（予測誤差）の絶対値和が、動きベクトル検出回路 50 から予測モード切り替え回路 52 と予測判定回路 54 に供給される。予測モード切り替え回路 52 と予測判定回路 54 は、この P ピクチャのマクロブロックの予測誤差の絶対値和に対応して、フレーム／フィールド予測モード、または画像内予測、前方予測、後方予測、もしくは両方向予測の予測モードを設定する。

【0074】

演算器 53 は、画像内予測モードが設定されたとき、スイッチ 53d を上述したように接点 a 側に切り替える。したがって、このデータは、I ピクチャのデータと同様に、DCT モード切り替え回路 55、DCT 回路 56、量子化回路 57、可変長符号化回路 58、および送信バッファ 59 を介して伝送路に伝送される。また、このデータは、逆量子化回路 6

0、IDCT回路61、および演算器62を介してフレームメモリ63の後方予測画像部63bに供給されて記憶される。

【0075】

また、前方予測モードが設定された場合、スイッチ53dが接点bに切り替えられるとともに、フレームメモリ63の前方予測画像部63aに記憶されている画像（いまの場合、Iピクチャの画像）データが読み出され、動き補償回路64により、動きベクトル検出回路50が出力する動きベクトルに対応して動き補償される。すなわち、動き補償回路64は、予測判定回路54より前方予測モードの設定が指令されたとき、前方予測画像部63aの読み出しアドレスを、動きベクトル検出回路50が、現在、出力しているマクロブロックの位置に対応する位置から動きベクトルに対応する分だけずらしてデータを読み出し、予測画像データを生成する。

10

【0076】

動き補償回路64より出力された予測画像データは、演算器53aに供給される。演算器53aは、予測モード切り替え回路52より供給された参照画像のマクロブロックのデータから、動き補償回路65より供給された、このマクロブロックに対応する予測画像データを減算し、その差分（予測誤差）を出力する。この差分データは、DCTモード切り替え回路55、DCT回路56、量子化回路57、可変長符号化回路58、および送信バッファ59を介して伝送路に伝送される。また、この差分データは、逆量子化回路60、およびIDCT回路61により局所的に復号され、演算器62に入力される。

【0077】

20

この演算器62にはまた、演算器53aに供給されている予測画像データと同一のデータが供給されている。演算器62は、IDCT回路61が出力する差分データに、動き補償回路64が出力する予測画像データを加算する。これにより、元の（復号した）Pピクチャの画像データが得られる。このPピクチャの画像データは、フレームメモリ63の後方予測画像部63bに供給されて記憶される。

【0078】

動きベクトル検出回路50は、このように、IピクチャとPピクチャのデータが前方予測画像部63aと後方予測画像部63bにそれぞれ記憶された後、次にBピクチャの処理を実行する。予測モード切り替え回路52と予測判定回路54は、マクロブロック単位でのフレーム間差分の絶対値和の大きさに対応して、フレーム／フィールドモードを設定し、また、予測モードを画像内予測モード、前方予測モード、後方予測モード、または両方向予測モードのいずれかに設定する。

30

【0079】

上述したように、画像内予測モードまたは前方予測モードの時、スイッチ53dは接点aまたはbに切り替えられる。このとき、Pピクチャにおける場合と同様の処理が行われ、データが伝送される。

【0080】

これに対して、後方予測モードまたは両方向予測モードが設定された時、スイッチ53dは、接点cまたはdにそれぞれ切り替えられる。

【0081】

40

スイッチ53dが接点cに切り替えられている後方予測モードの時、後方予測画像部63bに記憶されている画像（いまの場合、Pピクチャの画像）データが読み出され、動き補償回路64により、動きベクトル検出回路50が出力する動きベクトルに対応して動き補償される。すなわち、動き補償回路64は、予測判定回路54より後方予測モードの設定が指令されたとき、後方予測画像部63bの読み出しアドレスを、動きベクトル検出回路50が、現在、出力しているマクロブロックの位置に対応する位置から動きベクトルに対応する分だけずらしてデータを読み出し、予測画像データを生成する。

【0082】

動き補償回路64より出力された予測画像データは、演算器53bに供給される。演算器53bは、予測モード切り替え回路52より供給された参照画像のマクロブロックのデ

50

ータから、動き補償回路 6 4 より供給された予測画像データを減算し、その差分を出力する。この差分データは、DCTモード切り替え回路 5 5、DCT回路 5 6、量子化回路 5 7、可変長符号化回路 5 8、および送信バッファ 5 9 を介して伝送路に伝送される。

【0083】

スイッチ 5 3 d が接点 d に切り替えられている両方向予測モードの時、前方予測画像部 6 3 a に記憶されている画像（いまの場合、I ピクチャの画像）データと、後方予測画像部 6 3 b に記憶されている画像（いまの場合、P ピクチャの画像）データが読み出され、動き補償回路 6 4 により、動きベクトル検出回路 5 0 が出力する動きベクトルに対応して動き補償される。

【0084】

すなわち、動き補償回路 6 4 は、予測判定回路 5 4 より両方向予測モードの設定が指令されたとき、前方予測画像部 6 3 a と後方予測画像部 6 3 b の読み出しアドレスを、動きベクトル検出回路 5 0 がいま出力しているマクロブロックの位置に対応する位置から動きベクトル（この場合の動きベクトルは、前方予測画像用と後方予測画像用の 2 つとなる）に対応する分だけずらしてデータを読み出し、予測画像データを生成する。

【0085】

動き補償回路 6 4 より出力された予測画像データは、演算器 5 3 c に供給される。演算器 5 3 c は、動きベクトル検出回路 5 0 より供給された参照画像のマクロブロックのデータから、動き補償回路 6 4 より供給された予測画像データの平均値を減算し、その差分を出力する。この差分データは、DCTモード切り替え回路 5 5、DCT回路 5 6、量子化回路 5 7、可変長符号化回路 5 8、および送信バッファ 5 9 を介して伝送路に伝送される。

【0086】

B ピクチャの画像は、他の画像の予測画像とされることがないため、フレームメモリ 6 3 には記憶されない。

【0087】

なお、フレームメモリ 6 3 において、前方予測画像部 6 3 a と後方予測画像部 6 3 b は、必要に応じてバンク切り替えが行われ、所定の参照画像に対して、一方または他方に記憶されているものを、前方予測画像あるいは後方予測画像として切り替えて出力することができる。

【0088】

上述した説明においては、輝度ブロックを中心として説明をしたが、色差ブロックについても同様に、図 8 乃至図 11 に示すマクロブロックを単位として処理されて伝送される。なお、色差ブロックを処理する場合の動きベクトルは、対応する輝度ブロックの動きベクトルを垂直方向と水平方向に、それぞれ 1/2 にしたものが用いられる。

【0089】

図 12 は、図 5 のデコーダ 31 の構成を示すブロック図である。伝送路（記録媒体 3）を介して伝送された符号化された画像データは、図示せぬ受信回路で受信されたり、再生装置で再生され、受信バッファ 8 1 に一時記憶された後、復号回路 9 0 の可変長復号回路 8 2 に供給される。可変長復号回路 8 2 は、受信バッファ 8 1 より供給されたデータを可変長復号し、動きベクトル、予測モード、予測フラグ、および DCT フラグを動き補償回路 8 7 に出力し、量子化スケールを逆量子化回路 8 3 に出力するとともに、復号された画像データを逆量子化回路 8 3 に出力する。

【0090】

逆量子化回路 8 3 は、可変長復号回路 8 2 より供給された画像データを、同じく可変長復号回路 8 2 より供給された量子化スケールに従って逆量子化し、IDCT回路 8 4 に出力する。逆量子化回路 8 3 より出力されたデータ（DCT係数）は、IDCT回路 8 4 により、逆離散コサイン変換処理が施され、演算器 8 5 に供給される。

【0091】

IDCT回路 8 4 より演算器 8 5 に供給された画像データが、I ピクチャのデータである場合、そのデータは演算器 8 5 より出力され、演算器 8 5 に後に入力される画像データ（P

10

20

30

40

50

またはBピクチャのデータ)の予測画像データ生成のために、フレームメモリ86の前方予測画像部86aに供給されて記憶される。また、このデータは、フォーマット変換回路32(図5)に出力される。

【0092】

IDCT回路84より供給された画像データが、その1フレーム前の画像データを予測画像データとするPピクチャのデータであり、前方予測モードのデータである場合、フレームメモリ86の前方予測画像部86aに記憶されている、1フレーム前の画像データ(Iピクチャのデータ)が読み出され、動き補償回路87で可変長復号回路82より出力された動きベクトルに対応する動き補償が施される。そして、演算器85において、IDCT回路84より供給された画像データ(差分のデータ)と加算され、出力される。この加算されたデータ、すなわち、復号されたPピクチャのデータは、演算器85に後に入力される画像データ(BピクチャまたはPピクチャのデータ)の予測画像データ生成のために、フレームメモリ86の後方予測画像部86bに供給されて記憶される。

10

【0093】

Pピクチャのデータであっても、画像内予測モードのデータは、Iピクチャのデータと同様に、演算器85において処理は行われず、そのまま後方予測画像部86bに記憶される。

【0094】

このPピクチャは、次のBピクチャの次に表示されるべき画像であるため、この時点では、まだフォーマット変換回路32へ出力されない(上述したように、Bピクチャの後に入力されたPピクチャが、Bピクチャより先に処理され、伝送されている)。

20

【0095】

IDCT回路84より供給された画像データが、Bピクチャのデータである場合、可変長復号回路82より供給された予測モードに対応して、フレームメモリ86の前方予測画像部86aに記憶されているIピクチャの画像データ(前方予測モードの場合)、後方予測画像部86bに記憶されているPピクチャの画像データ(後方予測モードの場合)、または、その両方の画像データ(両方向予測モードの場合)が読み出され、動き補償回路87において、可変長復号回路82より出力された動きベクトルに対応する動き補償が施されて、予測画像が生成される。但し、動き補償を必要としない場合(画像内予測モードの場合)、予測画像は生成されない。

30

【0096】

このようにして、動き補償回路87で動き補償が施されたデータは、演算器85において、IDCT回路84の出力と加算される。この加算出力は、フォーマット変換回路32に出力される。

【0097】

ただし、この加算出力はBピクチャのデータであり、他の画像の予測画像生成のために利用されることがないため、フレームメモリ86には記憶されない。

【0098】

Bピクチャの画像が出力された後、後方予測画像部86bに記憶されているPピクチャの画像データが読み出され、動き補償回路87を介して演算器85に供給される。但し、このとき、動き補償は行われない。

40

【0099】

なお、このデコーダ31には、図5のエンコーダ18における予測モード切り替え回路52とDCTモード切り替え回路55に対応する回路が図示されていないが、これらの回路に対応する処理、すなわち、奇数フィールドと偶数フィールドのラインの信号が分離された構成を元の構成に必要なに応じて戻す処理は、動き補償回路87により実行される。

【0100】

また、上述した説明においては、輝度信号の処理について説明したが、色差信号の処理も同様に行われる。ただし、この場合の動きベクトルは、輝度信号用の動きベクトルを、垂直方向および水平方向に1/2にしたものが用いられる。

50

【0101】

図13は、符号化された画像の品質を示している。画像の品質(SNR:Signal to Noise Ratio)は、ピクチャタイプに対応して制御され、Iピクチャ、およびPピクチャは高品質とされ、Bピクチャは、I、Pピクチャに比べて劣る品質とされて伝送される。これは、人間の視覚特性を利用した手法であり、全ての画像品質を平均化するよりも、品質を振動させたほうが視覚上の画質が良くなるためである。このピクチャタイプに対応した画質の制御は、図7の量子化回路57により実行される。

【0102】

図14及び図15は、本発明を適用したトランスコーダ101の構成を示しており、図15は、図14のさらに詳細な構成を示している。このトランスコーダ101は、復号装置102に入力された符号化ビデオビットストリーム(encoded video bit stream)のGOP構造及びビットレートを、オペレータの所望するGOP構造及びビットレートに変換する。このトランスコーダ101の機能を説明するために、図15には図示されていないが、このトランスコーダ101の前段に、このトランスコーダ101とほぼ同様の機能を有した3つのトランスコーダが接続されているものとする。つまり、ビットストリームのGOP構造及びビットレートをさまざまに変更するために、第1のトランスコーダ、第2のトランスコーダ、および第3のトランスコーダが順に直列に接続され、その第3のトランスコーダの後ろに、この図15に示された第4のトランスコーダが接続されているものとする。

【0103】

本発明の以下の説明において、この第1のトランスコーダにおいて行われた符号化処理を第1世代の符号化処理と定義し、第1のトランスコーダの後ろに接続された第2のトランスコーダにおいて行われた符号化処理を第2世代の符号化処理と定義し、第2のトランスコーダの後ろに接続された第3のトランスコーダにおいて行われた符号化処理を第3世代の符号化処理と定義し、第3のトランスコーダの後ろに接続された第4のトランスコーダ(図15に示されたトランスコーダ101)において行われる符号化処理を第4世代の符号化処理または現在の符号化処理と定義することにする。

【0104】

また、第1世代の符号化処理において生成された符号化パラメータを第1世代の符号化パラメータと呼び、第2世代の符号化処理において生成された符号化パラメータを第2世代の符号化パラメータと呼び、第3世代の符号化処理において生成された符号化パラメータを第3世代の符号化パラメータと呼び、第4世代の符号化処理において生成された符号化パラメータを第4世代の符号化パラメータまたは現在の符号化パラメータと呼ぶことにする。

【0105】

まず、この図15に示されたトランスコーダ101に供給される符号化ビデオストリームST(3rd)について説明する。ST(3rd)は、このトランスコーダ101の前段に設けられている第3のトランスコーダにおける第3世代の符号化処理において生成された第3世代の符号化ストリームであることを表わしている。この第3世代の符号化処理において生成された符号化ビデオストリームST(3rd)には、第3の符号化処理において生成された第3世代の符号化パラメータが、この符号化ビデオストリームST(3rd)のシーケンス層、GOP層、ピクチャ層、スライス層、及びマクロブロック層に、sequence_header()関数、sequence_extension()関数、group_of_pictures_header()関数、picture_header()関数、picture_coding_extension()関数、picture_data()関数、slice()関数、及びmacroblock()関数として記述されている。このように第3の符号化処理によって生成された第3の符号化ストリームに、第3の符号化処理において使用した第3の符号化パラメータを記述することはMPEG2規格において定義されていることであって、何ら新規性は無い。

【0106】

本発明のトランスコーダ101におけるユニークな点は、この第3の符号化ストリームST(3rd)中に、第3の符号化パラメータを記述するだけでなく、第1世代及び第2世

10

20

30

40

50

代の符号化処理において生成された第1世代及び第2世代の符号化パラメータも記述されているという点である。

【0107】

具体的には、この第1世代及び第2世代の符号化パラメータは、第3世代の符号化ビデオストリームS T (3rd)のピクチャ層のユーザデータエリアに、ヒストリストリームhistory_stream()として記述されている。本発明においては、第3世代の符号化ビデオストリームS T (3rd)のピクチャ層のユーザデータエリアに記述されているヒストリストリームを、「ヒストリ情報」、または「履歴情報」と呼び、このヒストリストリームとして記述されている符号化パラメータを「ヒストリパラメータ」、または「履歴パラメータ」と呼んでいる。

10

【0108】

また別の呼び方として、第3世代の符号化ストリームS T (3rd)に記述されている第3世代の符号化パラメータを「現在の符号化パラメータ」と呼んだ場合には、第3世代の符号化処理からみて第1世代及び第2世代の符号化処理は、過去に行なわれた符号化処理であるので、第3世代の符号化ストリームS T (3rd)のピクチャ層のユーザデータエリアに記述されているヒストリストリームとして記述されている符号化パラメータを「過去の符号化パラメータ」とも呼んでいる。

【0109】

このように、この第3の符号化ストリームS T (3rd)中に、第3の符号化パラメータを記述するだけでなく、第1世代及び第2世代の符号化処理において生成された第1世代及び第2世代の符号化パラメータを記述する理由は、トランスコーディング処理によって符号化ストリームのGOP構造やビットレートの変更を繰り返したとしても、画質劣化を防止することができるからである。

20

【0110】

例えば、あるピクチャを第1世代の符号化処理においてPピクチャとして符号化し、第1世代の符号化ストリームのGOP構造を変更するために、第2世代の符号化処理においてそのピクチャをBピクチャとして符号化し、第2世代の符号化ストリームのGOP構造をさらに変更するために、第3世代の符号化処理において、再度そのピクチャをPピクチャとして符号化することが考えられる。MPEG規格に基づく符号化処理及び復号処理は100%可逆の処理ではないので、符号化及び復号処理を繰り返す毎に画質が劣化していくことは知られている。

30

【0111】

このような場合に、第3の世代の符号化処理において、量子化スケール、動きベクトル、予測モードなどの符号化パラメータをもう一度計算するのではなくて、第1世代の符号化処理において生成された量子化スケール、動きベクトル、予測モードなどの符号化パラメータを再利用する。第3世代の符号化処理によって新しく生成された量子化スケール、動きベクトル、予測モードなどの符号化パラメータよりも、第1世代の符号化処理によって新しく生成された量子化スケール、動きベクトル、予測モードなどの符号化パラメータの方が、明らかに精度が良いので、この第1世代のパラメータを再利用することによって、符号化及び復号処理を繰り返したとしても画質劣化を少なくすることができる。

40

【0112】

上述した本発明にかかる処理を説明するために、図15に示された第4世代のトランスコーダ101の処理を例に挙げてより詳しく説明する。

【0113】

復号装置102は、第3世代の符号化ビットストリームS T (3rd)に含まれている符号化ビデオを第3世代の符号化パラメータを使用して復号し、復号されたベースバンドのデジタルビデオデータを生成するための装置である。さらに、復号装置102は、第3世代の符号化ビットストリームS T (3rd)のピクチャ層のユーザデータエリアにヒストリストリームとして記述されている第1世代及び第2世代の符号化パラメータをデコードするための装置でもある。

50

【0114】

具体的には、図16に示されているように、復号装置102は、図5の復号装置2のデコーダ31（図12）と基本的に同様の構成とされ、供給されたビットストリームをバッファリングするための受信バッファ81、符号化ビットストリームを可変長復号するための可変長復号回路112、可変長復号されたデータを可変長復号回路112から供給された量子化スケールに従って逆量子化する逆量子化回路83、逆量子化されたDCT係数を逆離散コサイン変換するIDCT回路84、及び動き補償処理を行うための演算器85、フレームメモリ86及び動き補償回路87を備えている。

【0115】

可変長復号回路112は、第3世代の符号化ビットストリームST（3rd）を復号処理するために、この第3世代の符号化ビットストリームST（3rd）のピクチャ層、スライス層及びマクロブロック層に記述されている第3世代の符号化パラメータを抽出する。たとえば、この可変長復号回路112において抽出される第3世代の符号化パラメータは、ピクチャタイプを示すpicture_coding_type、量子化スケールステップサイズを示すquantiser_scale_code、予測モードを示すmacroblock_type、動きベクトルを示すmotion_vector、Frame予測モードかField予測モードかを示すframe/field_motion_type、及びFrameDCTモードかFieldDCTモードかを示すdct_type等である。この可変長復号回路112において抽出されたquantiser_scale_codeは、逆量子化回路83に供給され、picture_coding_type、quantiser_scale_code、macroblock_type、motion_vector、frame/field_motion_type、dct_type等のパラメータは、動き補償回路87に供給される。

【0116】

可変長復号回路112は、第3世代の符号化ビットストリームST（3rd）を復号処理するために必要なこれらの符号化パラメータだけではなく、後段の第5世代のトランスコーダに第3世代のヒストリ情報として伝送されるべき符号化パラメータを、第3世代の符号化ビットストリームST（3rd）のシーケンス層、GOP層、ピクチャ層、スライス層、及びマクロブロック層から抽出する。もちろん、第3世代の復号処理に使用されたpicture_coding_type、quantiser_scale_code、macroblock_type、motion_vector、frame/field_motion_type、dct_type等の第3世代の符号化パラメータは、この第3世代のヒストリ情報に含まれている。ヒストリ情報としてどのような符号化パラメータを抽出するかについては、伝送容量などに応じてオペレータやホストコンピュータ側からあらかじめ設定されている。

【0117】

さらに、可変長復号回路112は、第3世代の符号化ビットストリームST（3rd）のピクチャ層のユーザデータエリアに記述されているユーザデータを抽出し、そのユーザデータをヒストリデコーディング装置104に供給する。

【0118】

このヒストリデコーディング装置104は、第3世代の符号化ビットストリームST（3rd）のピクチャ層に記述されていたユーザデータから、ヒストリ情報として記述されている第1世代の符号化パラメータ及び第2世代の符号化パラメータ（直前の世代よりさらに前の世代の符号化パラメータ）を抽出するための回路である。具体的には、ヒストリデコーディング装置104は、受け取ったユーザデータのシンタックスを解析することによって、ユーザデータの中に記述されている固有のHistory_Data_Idを検出し、これによって、converted_history_stream()を抽出することができる。さらに、ヒストリデコーディング装置104は、converted_history_stream()中にある所定間隔に挿入されている1ビットのマーカビット(marker_bit)を取りさることによって、history_stream()を得、そして、そのhistory_stream()のシンタックスを解析することによって、history_stream()中に記述されている第1世代及び第2世代の符号化パラメータを得ることができる。このヒストリデコーディング装置104の詳しい動作については、後述する。

【0119】

ヒストリ情報多重化装置103は、第1世代、第2世代及び第3世代の符号化パラメータ

10

20

30

40

50

タを、第4世代の符号化処理を行う符号化装置106に供給するために、復号装置102においてデコードされたベースバンドのビデオデータに、これらの第1世代、第2世代及び第3世代の符号化パラメータを多重化するための回路である。具体的には、ヒストリ情報多重化装置103は、復号装置102の演算器85から出力されたベースバンドのビデオデータ、復号装置102の可変長復号装置112から出力された第3世代の符号化パラメータ、並びに、ヒストリデコーディング装置104から出力された第1世代の符号化パラメータと第2世代の符号化パラメータとを受け取り、このベースバンドのビデオデータに、これらの第1世代、第2世代及び第3世代の符号化パラメータを多重化する。第1世代、第2世代及び第3世代の符号化パラメータが多重化されたベースバンドのビデオデータは、伝送ケーブルを介してヒストリ情報分離装置105に供給される。

10

【0120】

次に、これらの第1世代、第2世代及び第3世代の符号化パラメータのベースバンドビデオデータへの多重化の方法について、図17及び図18を参照して説明する。図17は、MPEG規格において定義されている、16ピクセル×16ピクセルからなる1つのマクロブロックを示している。この16ピクセル×16ピクセルのマクロブロックは、輝度信号に関しては4つの8ピクセル×8ピクセルからなるサブブロック（Y[0]，[1]，[2]及びY[3]）と、色差信号に関しては4つの8ピクセル×8ピクセルからなるサブブロック（Cr[0]，r[1]，b[0]，及びCb[1]）から構成されている。図18は、ビデオデータのあるフォーマットを表している。このフォーマットは、ITU勧告-RDT601において定義されているフォーマットであって、放送業界において使用されている所謂「D1フォーマット」をあらわしている。このD1フォーマットは、10ビットのビデオデータを伝送するためのフォーマットとして規格化されたので、ビデオデータの1ピクセルを10ビットで表現できるようになっている。

20

【0121】

MPEG規格によってデコードされたベースバンドのビデオデータは8ビットであるので、本発明のトランスコーダにおいては、図18に示したように、D1フォーマットの10ビットのうち上位8ビット（D9乃至D2）を使用して、MPEG規格にもとづいてデコードされたベースバンドのビデオデータを伝送するようにしている。このように、復号された8ビットのビデオデータをD1フォーマットに書き込むと、下位2ビット（D1とD0）は、空きビット（unallocated bits）となる。本発明のトランスコーダではこの空きエリア（unallocated area）を利用して、ヒストリ情報を伝送するようにしている。

30

【0122】

この図18に記載されたデータブロックは、各サブブロック（Y[0]，Y[1]，Y[2]，Y[3]，Cr[0]，Cr[1]，Cb[0]，Cb[1]）における1ピクセルを伝送するためのデータブロックであるので、1マクロブロックのデータを伝送するためには、この図18に示されているデータブロックが64個伝送される。下位2ビット（D1とD0）を使用すれば、1マクロブロックのビデオデータに対して、合計で1024（＝16×64）ビットのヒストリ情報を伝送できる。従って、1世代分のヒストリ情報は、256ビットとなるように生成されているので、過去の4（＝1024／256）世代分のヒストリ情報を1マクロブロックのビデオデータに対して重畳することができる。図18に示した例では、第1世代のヒストリ情報、第2世代のヒストリ情報、並びに、第3世代のヒストリ情報が重畳されている。

40

【0123】

ヒストリ情報分離装置105は、D1フォーマットとして伝送されたデータの上位8ビットから、ベースバンドビデオデータを抽出し、下位2ビットからヒストリ情報を抽出するための回路である。図15に示した例では、ヒストリ情報分離装置105は、伝送データからベースバンドのビデオデータを抽出して、そのビデオデータを符号化装置106に供給するとともに、伝送データから第1世代、第2世代及び第3世代のヒストリ情報を抽出して、符号化装置106とヒストリエンコーディング装置107にそれぞれ供給する。

【0124】

50

符号化装置 106 は、ヒストリ情報分離装置 105 から供給されたベースバンドのビデオデータを、オペレータまたはホストコンピュータから指定された GOP 構造及びビットレートを有するビットストリームになるように符号化するための装置である。なお、GOP 構造を変更するとは、たとえば、GOP に含まれるピクチャの数、I ピクチャと I ピクチャの間に存在する P ピクチャの数、及び I ピクチャと P ピクチャ（または I ピクチャ）の間に存在する B ピクチャの数を変更することを意味する。

【0125】

図 15 に示された例では、供給されたベースバンドのビデオデータには、第 1 世代、第 2 世代及び第 3 世代のヒストリ情報が重畳されているので、この符号化装置 106 は、再符号化処理による画質劣化が少なくなるように、これらのヒストリ情報を選択的に再利用して第 4 世代の符号化処理を行う。

10

【0126】

図 19 は、この符号化装置 106 に設けられている符号化装置 106 の具体的な構成を示している図である。この符号化装置 106 は、基本的には、図 7 に示したエンコーダ 18 と同様に構成され、動きベクトル検出回路 50、フレーム／フィールド予測モード切り替え回路 52、演算器 53、DCT モード切り替え回路 55、DCT 回路 56、量子化回路 57、可変長符号化回路 58、伝送バッファ 59、逆量子化回路 60、逆 DCT 回路 61、演算器 62、フレームメモリ 63、並びに動き補償回路 64 を備えている。これらの、各回路の機能は、図 7 において説明したエンコーダ 18 における場合の機能とほぼ同じであるので、その説明は省略する。以下に、この符号化装置 106 と、図 7 において説明したエンコーダ 18 との異なる点を中心に説明する。

20

【0127】

この符号化装置 106 は、上述した各回路の動作及び機能を制御するためのコントローラ 70 を有している。このコントローラ 70 は、オペレータまたはホストコンピュータから GOP 構造に関するインストラクションを受け取って、その GOP 構造に対応するように各ピクチャのピクチャタイプを決定する。また、このコントローラ 70 は、オペレータまたはホストコンピュータからターゲットビットレート of 情報を受け取り、この符号化装置 106 から出力されるビットレートがこの指定されたターゲットビットレートになるように、量子化回路 57 を制御する。

【0128】

さらに、このコントローラ 70 は、ヒストリ情報分離装置 105 から出力された複数世代のヒストリ情報を受け取り、これらのヒストリ情報を再利用して参照ピクチャの符号化処理を行う。以下に詳しく説明する。

30

【0129】

まず、このコントローラ 70 は、オペレータによって指定された GOP 構造から決定された参照ピクチャのピクチャタイプと、ヒストリ情報に含まれるピクチャタイプが一致するか否かを判断する。つまり、指定されたピクチャタイプと同じピクチャタイプでこの参照ピクチャが過去において符号化されたことがあるか否かを判断する。

【0130】

図 15 に示された例をあげてよりわかりやすく説明するのであれば、このコントローラ 70 は、第 4 世代の符号化処理としてこの参照ピクチャにアサインされたピクチャタイプが、第 1 世代の符号化処理におけるこの参照ピクチャのピクチャタイプ、第 2 世代の符号化処理におけるこの参照ピクチャのピクチャタイプ、または第 3 世代の符号化処理におけるこの参照ピクチャのピクチャタイプのいずれかと一致するか否かを判断する。

40

【0131】

もし、第 4 世代の符号化処理としてこの参照ピクチャに指定されたピクチャタイプが、過去の符号化処理におけるどのピクチャタイプとも一致しないのであれば、このコントローラ 70 は、「通常符号化処理」を行う。つまり、この場合には、第 1 世代、第 2 世代または第 3 世代のどの世代の符号化処理においても、第 4 世代の符号化処理としてアサインされたピクチャタイプで、この参照ピクチャが符号化処理されたことがないということに

50

なる。一方、もし、第4世代の符号化処理としてこの参照ピクチャに指定されたピクチャタイプが、過去の符号化処理におけるいずれかのピクチャタイプと一致するのであれば、このコントローラ70は、「パラメータ再利用符号化処理」を行う。つまり、この場合には、第1世代、第2世代または第3世代のいずれかの世代の符号化処理において、第4世代の符号化処理としてアサインされたピクチャタイプで、この参照ピクチャが符号化処理されたことがあるということになる。

【0132】

まず、最初にコントローラ70の通常符号化処理について説明する。

【0133】

動きベクトル検出回路50は、フレーム予測モードまたはフィールド予測モードのどちらが選択されるべきかを判断するために、フレーム予測モードにおける予測誤差とフィールド予測モードにおける予測誤差をそれぞれ検出し、その予測誤差の値をコントローラ70に供給する。コントローラ70は、それらの予測誤差の値を比較し、その予測誤差の値が小さい方の予測モードを選択する。予測モード切り替え回路52は、コントローラ70によって選択された予測モードに対応するように信号処理を行い、それを演算器53に供給する。

10

【0134】

具体的には、予測モード切り替え回路52は、フレーム予測モードが選択された場合には、図8を参照して説明したように、輝度信号に関しては、入力された状態のまま演算器53に出力するように信号処理を行い、色差信号に関しては、奇数フィールドラインと偶数フィールドラインとが混在するように信号処理する。一方、フィールド予測モードが選択された場合には、図9を参照して説明したように、輝度信号に関しては、輝度ブロックY[1]とY[2]を奇数フィールドラインで構成し、輝度ブロックY[3]とY[4]を偶数フィールドラインで構成するように信号処理し、色差信号に関しては、上4ラインを奇数フィールドラインで構成し、下4ラインを偶数フィールドラインで構成するように信号処理する。

20

【0135】

さらに、動きベクトル検出回路50は、画像内予測モード、前方予測モード、後方予測モード、または両方向予測モードのうちのいずれの予測モードを選択するかを決定するために、各予測モードにおける予測誤差を生成し、各予測モードにおける予測誤差をコントローラ70にそれぞれ供給する。コントローラ70は、前方予測、後方予測および両方向予測の予測誤差のうちの最も小さいものを、インタ予測の予測誤差として選択する。さらに、このインタ予測の予測誤差と、画像内予測の予測誤差とを比較し、その小さい方を選択し、この選択した予測誤差に対応するモードを予測モードとして選択する。すなわち、画像内予測の予測誤差の方が小さければ、画像内予測モードが設定される。インタ予測の予測誤差の方が小さければ、前方予測、後方予測または両方向予測モードのうちの対応する予測誤差が最も小さかったモードが設定される。コントローラ70は、選択した予測モードに対応するように、演算器53及び動き補償回路64を制御する。

30

【0136】

DCTモード切り替え回路55は、フレームDCTモードまたはフィールドDCTモードのいずれかを選択するために、4個の輝度ブロックのデータを、奇数フィールドラインと偶数フィールドラインが混在するような信号形態（フレームDCTモード）に変換するとともに、奇数フィールドラインと偶数フィールドラインが分離された信号形態（フィールドDCTモード）に変換して、それぞれの信号をDCT回路56に供給する。DCT回路56は、奇数フィールドと偶数フィールドを混在してDCT処理した場合における符号化効率と、奇数フィールドと偶数フィールドを分離した状態においてDCT処理した場合の符号化効率を計算し、その結果をコントローラ70に供給する。コントローラ70は、DCT回路56から供給されたそれぞれの符号化効率を比較し、符号化効率の良い方のDCTモードを選択し、その選択したDCTモードとなるようにDCTモード切り替え回路55を制御する。

40

50

【0137】

コントローラ70は、オペレータまたはホストコンピュータから供給された目標ビットレートを示すターゲットビットレートと、送信バッファ59にバッファリングされているビット量を示す信号、つまり、バッファ残量を示す信号を受け取り、このターゲットビットレートとバッファ残量に基づいて、量子化回路57の量子化ステップサイズをコントロールするためのfeedback_q_scale_codeを生成する。このfeedback_q_scale_codeは、この送信バッファ59がオーバーフローまたはアンダーフローしないように、この送信バッファ59のバッファ残量に応じて生成される制御信号であって、また、送信バッファ59から出力されるビットストリームのビットレートが、ターゲットビットレートになるように制御する信号でもある。

10

【0138】

具体的には、例えば、送信バッファ59にバッファリングされているビット量が少なくなってしまう場合には、次に符号化するピクチャの発生ビット量が増えるように、量子化ステップサイズを小さくし、一方、送信バッファ59にバッファリングされているビット量が多くなってしまう場合には、次に符号化するピクチャの発生ビット量が少なくなるように、量子化ステップサイズを大きくする。なお、feedback_q_scale_codeと量子化ステップサイズは比例し、feedback_q_scale_codeを大きくすると、量子化ステップサイズは大きくなり、feedback_q_scale_codeを小さくすると、量子化ステップサイズは小さくなる。

【0139】

20

次に、このトランスコーダ101の特徴の1つでもある、パラメータ再利用符号化処理について説明する。この処理をより分かりやすく説明するために、参照ピクチャは、第1世代の符号化処理においてPピクチャとして符号化され、第2世代の符号化処理においてIピクチャとして符号化処理され、第3世代の符号化処理においてBピクチャとして符号化されていたものとし、今回の第4世代の符号化処理において、この参照ピクチャをPピクチャとして符号化しなければならないものとする。

【0140】

この場合には、第4世代のピクチャタイプとしてアサインされたピクチャタイプと同じピクチャタイプ（Iピクチャ）で、この参照ピクチャは第1世代の符号化処理において符号化されているので、コントローラ70は、供給されたビデオデータから符号化パラメータを新しく作成するのではなく、第1世代の符号化パラメータを使用して符号化処理を行う。この第4の符号化処理において再利用する符号化パラメータは、代表的なパラメータとしては、量子化スケールステップサイズを示すquantiser_scale_code、予測方向モードを示すmacroblock_type、動きベクトルを示すmotion_vector、Frame予測モードかField予測モードかを示すframe/field_motion_type、及びFrame DCTモードかField DCTモードかを示すdct_type等である。

30

【0141】

コントローラ70は、履歴情報として伝送されたすべての符号化パラメータを再利用するわけではなく、再利用した方が望ましいと想定される上述したような符号化パラメータについては再利用し、再利用しない方が望ましいと考えられる符号化パラメータについては、新しく生成する。

40

【0142】

次に、この符号化パラメータ再利用符号化処理について、上述した通常符号化処理と異なる点を中心に説明する。

【0143】

動きベクトル検出回路50は、上述した通常符号化処理においては、参照ピクチャの動きベクトルの検出を行ったが、このパラメータ再利用符号化処理においては、動きベクトルmotion_vectorの検出処理は行わずに、第1世代の履歴情報として供給された動きベクトルmotion_vectorを再利用する。その理由について説明する。

【0144】

50

第3世代の符号化ストリームを復号したベースバンドのビデオデータは、少なくとも3回の復号及び符号化処理が行われているので、オリジナルビデオデータに比べると、明らかに画質が劣化している。画質が劣化しているビデオデータから動きベクトルを検出したとしても、正確な動きベクトルは検出できない。つまり、第4世代の符号化処理において検出された動きベクトルよりも、第1世代のヒストリ情報として供給されている動きベクトルの方が、明らかに、精度の高い動きベクトルである。つまり、第1世代の符号化パラメータとして伝送された動きベクトルを再利用することによって、第4世代の符号化処理を行ったとしても画質が劣化しない。コントローラ70は、この第1世代のヒストリ情報として供給された動きベクトルmotion_vectorを、第4世代の符号化処理において符号化されるこの参照ピクチャの動きベクトル情報として、動き補償回路64及び可変長符号化回路58に供給する。

10

【0145】

さらに、動きベクトル検出回路50は、フレーム予測モードとフィールド予測モードのどちらが選択されるかを判断するために、フレーム予測モードにおける予測誤差とフィールド予測モードにおける予測誤差をそれぞれ検出したが、このパラメータ再利用符号化処理においては、このフレーム予測モードにおける予測誤差とフィールド予測モードにおける予測誤差を検出する処理は行わずに、第1世代のヒストリ情報として供給されているFrame予測モードかField予測モードかを示すframe/field_motion_typeを再利用する。なぜなら、第4世代の符号化処理において検出された各予測モードにおける予測誤差よりも、第1世代において検出された各予測モードにおける予測誤差の方が精度が高いので、精度の高い予測誤差によって決定された予測モードを選択した方がより最適な符号化処理が行うことができるからである。

20

【0146】

具体的には、コントローラ70は、この第1世代のヒストリ情報として供給されているframe/field_motion_typeに対応する制御信号を予測モード切り替え回路52に供給し、予測モード切り替え回路52は、この再利用されたframe/field_motion_typeに対応した信号処理を行う。

【0147】

さらには、動きベクトル検出回路50は、通常符号化処理においては、画像内予測モード、前方予測モード、後方予測モード、または両方向予測モードのうちのいずれの予測モード（以下、この予測モードを、予測方向モードとも称する）を選択するかを決定するために、各予測方向モードにおける予測誤差を計算していたが、このパラメータ再利用符号化処理においては、各予測方向モードにおける予測誤差の計算は行わず、第1世代のヒストリ情報として供給されたmacroblock_typeに基づいて予測方向モードを決定する。なぜなら、第4世代の符号化処理における各予測方向モードにおける予測誤差よりも、第1世代の符号化処理における各予測方向モードにおける予測誤差の方がより精度が高いので、より精度の高い予測誤差によって決定された予測方向モードを選択した方が、より効率の良い符号化処理が行えるからである。具体的には、コントローラ70は、第1世代のヒストリ情報に含まれているmacroblock_typeによって示される予測方向モードを選択し、その選択した予測方向モードに対応するように、演算器53及び動き補償回路64をコントロールする。

30

40

【0148】

DCTモード切り替え回路55は、通常符号化処理においては、フレームDCTモードの符号化効率と、フィールドDCTモードの符号化効率を比較するために、フレームDCTモードの信号形態に変換した信号と、フィールドDCTモードの信号形態に変換した信号の両方をDCT回路56に供給していたが、このパラメータ再利用符号化処理では、フレームDCTモードの信号形態に変換した信号と、フィールドDCTモードの信号形態に変換した信号の両方を生成する処理は行っておらず、第1世代のヒストリ情報に含まれているdct_typeによって示されたDCTモードに対応する処理のみを行っている。具体的には、コントローラ70は、第1世代のヒストリ情報に含まれているdct_typeを再利用し、DCTモード

50

切り替え回路 55 がこのdct_typeによって示される D C T モードに対応した信号処理を行うように、D C T モード切り替え回路 55 をコントロールする。

【0149】

コントローラ 70 は、通常符号化処理では、オペレータによって指定されたターゲットビットレートと送信バッファ残量に基づいて、量子化回路 57 の量子化ステップサイズをコントロールしていたが、このパラメータ再利用符号化処理では、ターゲットビットレート、送信バッファ残量及びヒストリ情報に含まれている過去の量子化スケールに基づいて、量子化回路 57 の量子化ステップサイズをコントロールする。なお、以下の説明において、ヒストリ情報に含まれている過去の量子化スケールをhistory_q_scale_code と記述することにする。また、後述するヒストリストリームにおいては、この量子化スケールを、quantiser_scale_codeと記述している。

10

【0150】

まず、コントローラ 70 は、通常符号化処理と同じように、現在の量子化スケールfeedback_q_scale_code を生成する。このfeedback_q_scale_codeは、この送信バッファ 59 がオーバーフロー及びアンダーフローしないように、この送信バッファ 59 のバッファ残量に応じて決定される値である。続いて、第 1 世代のヒストリストリームに含まれている過去の量子化スケールhistory_q_scale_codeの値と、この現在の量子化スケールfeedback_q_scale_codeの値を比較し、どちらの量子化スケールの方が大きいかを判断する。量子化スケールが大きいとは、量子化ステップが大きいことを意味する。もし、現在の量子化スケールfeedback_q_scale_codeが、過去の量子化スケールhistory_q_scale_codeよりも大きいのであれば、コントローラ 70 は、この現在の量子化スケールfeedback_q_scale_codeを量子化回路 57 に供給する。一方、過去の量子化スケールhistory_q_scale_codeが、現在の量子化スケールfeedback_q_scale_codeよりも大きいのであれば、コントローラ 70 は、この過去の量子化スケールhistory_q_scale_codeを量子化回路 57 に供給する。

20

【0151】

つまり、コントローラ 70 は、ヒストリ情報に含まれている複数の過去の量子化スケールと、送信バッファの残量から計算された現在の量子化スケールの中で、もっとも大きい量子化スケールコードを選択する。また、別の言葉で説明するのであれば、コントローラ 70 は、過去（第 1、第 2 及び第 3 世代）の符号化処理における量子化ステップまたは現在（第 4 世代）の符号化処理において使用された量子化ステップの中で、もっとも大きい量子化ステップを使用して量子化を行うように量子化回路 57 を制御する。この理由を以下に説明する。

30

【0152】

たとえば、第 3 世代の符号化処理において生成されたストリームのビットレートが 4 [Mbps]であって、この第 4 世代の符号化処理を行う符号化装置 106 に対して設定されたターゲットビットレートが 1.5 [Mbps]であったとする。このときに、ターゲットビットレートが上がっているので、単純に量子化ステップを小さくすれば良いかというと、実際にはそうではない。過去の符号化処理において大きい量子化ステップで符号化処理されたピクチャを、現在の符号化処理において、量子化ステップを小さくして符号化処理を行ったとしても、このピクチャの画質は向上することはない。つまり、過去の符号化処理における量子化ステップよりも小さい量子化ステップで符号化することは、単にビット量が増えるだけであって、画質を向上させることにはならない。よって、過去（第 1、第 2 及び第 3 世代）の符号化処理における量子化ステップまたは現在（第 4 世代）の符号化処理において使用された量子化ステップの中で、もっとも大きい量子化ステップを使用して量子化を行うと、もっとも効率の良い符号化処理が行える。

40

【0153】

次に、図 15 におけるヒストリデコーディング装置 104 とヒストリエンコーディング装置 107 についてさらに説明する。尚、図 15 においては、このヒストリデコーディング装置 104 が、復号化装置 102 とは、別の回路又は装置のように表現されているが、これはヒストリデコーディング装置 104 の機能及び構成をよりわかりやすく説明するた

50

めに、復号化装置102とは別のブロックのように表現しただけであって、実際には、ヒストリデコーディング装置104の処理は、復号化装置102の可変長復号化回路及び復号化制御回路（復号化コントローラ）の中で行われている処理である。同様に、図15においては、このヒストリエンコーディング装置107が、符号化装置106とは、別の回路又は装置のように表現されているが、これはヒストリエンコーディング装置107の機能及び構成をよりわかりやすく説明するために、符号化装置106とは別のブロックのように表現しただけであって、実際には、ヒストリエンコーディング装置107の処理は、符号化装置102の可変長符号化回路及び符号化制御回路（符号化コントローラ）の中で行われている処理である。

【0154】

10

図15に示すように、ヒストリデコーディング装置104は、復号装置102より供給されるユーザデータをデコードするユーザデータデコーダ201、ユーザデータデコーダ201の出力を変換するコンバータ202、およびコンバータ202の出力から履歴情報を再生するヒストリVLD203により構成されている。

【0155】

また、ヒストリエンコーディング装置107は、ヒストリ情報分離装置105より供給される3世代分の符号化パラメータをフォーマット化するヒストリVLC211、ヒストリVLC211の出力を変換するコンバータ212、コンバータ212の出力をユーザデータのフォーマットにフォーマットするユーザデータフォーマッタ213により構成されている。

20

【0156】

ユーザデータデコーダ201は、復号装置102より供給されるユーザデータをデコードして、コンバータ202に出力する。詳細は図51を参照して後述するが、ユーザデータ（`user_data()`）は、`user_data_start_code`と`user_data`からなり、MPEG規格においては`user_data`の中に、連続する23ビットの“0”（`start_code`と同一のコード）を発生させることを禁止している。これは、そのデータが、`start_code`として誤検出されるのを防止するためである。履歴情報（`history_stream()`）は、ユーザデータエリアに（MPEG規格の`user_data`の一種として）記述され、その中には、このような連続する23ビット以上の“0”が存在することがあり得るので、これを、連続する23ビット以上の“0”が発生しないように、所定のタイミングで“1”を挿入処理して、`converted_history_stream()`（後述する図38）に変換する必要がある。この変換を行うのは、ヒストリエンコーディング装置107のコンバータ212である。ヒストリデコーディング装置104のコンバータ202は、このコンバータ212と逆の変換処理を行う（連続する23ビット以上の“0”を発生させないために挿入された“1”を除去する）ものである。

30

【0157】

ヒストリVLD203は、コンバータ202の出力から履歴情報（いまの場合、第1世代の符号化パラメータと第2世代の符号化パラメータ）を生成し、ヒストリ情報多重化装置103に出力する。

【0158】

一方、ヒストリエンコーディング装置107においては、ヒストリVLC211がヒストリ情報分離装置105より供給される3世代分の（第1世代、第2世代、および第3世代の）符号化パラメータを履歴情報のフォーマットに変換する。このフォーマットには、固定長のもの（後述する図40乃至図46）と、可変長のもの（後述する図47以降）とがある。これらの詳細については後述する。

40

【0159】

ヒストリVLC211により、フォーマット化された履歴情報は、コンバータ212において、`converted_history_stream()`に変換される。これは、上述したように、`user_data()`の`start_code`が誤検出されないようにするための処理である。すなわち、履歴情報内には連続する23ビット以上の“0”が存在するが、`user_data`中には連続する23ビット以上の“0”を配置することができないので、この禁止項目に触れないようにコンバータ

50

212によりデータを変換する（“1”を所定のタイミングで挿入する）のである。

【0160】

ユーザデータフォーマッタ213は、コンバータ212より供給されるconverted_history_stream()に、後述する図38に基づいて、History_Data_IDを付加し、さらに、user_data_stream_codeを付加して、video stream中に挿入できるMPEG規格のuser_dataを生成し、符号化装置106に出力する。

【0161】

図20は、ヒストリVLC211の構成例を表している。その符号語変換器301と符号長変換器305には、符号化パラメータ（今回、履歴情報として伝送する符号化パラメータ）（項目データ）と、この符号化パラメータを配置するストリームを特定する情報（例えば、シンタックスの名称（例えば、後述するsequence_headerの名称））（項目NO.）が、ヒストリ情報分離装置105から供給されている。符号語変換器301は、入力された符号化パラメータを、指示されたシンタックスに対応する符号語に変換し、バレルシフタ302に出力する。バレルシフタ302は、符号語変換器301より入力された符号語を、アドレス発生回路306より供給されるシフト量に対応する分だけシフトし、バイト単位の符号語として、スイッチ303に出力する。アドレス発生回路306が出力するビットセレクト信号により切り換えられるスイッチ303は、ビット分設けられており、バレルシフタ302より供給される符号語を、RAM304に供給し、記憶させる。このときの書き込みアドレスは、アドレス発生回路306から指定される。また、アドレス発生回路306から読み出しアドレスが指定されたとき、RAM304に記憶されているデータ（符号語）が読み出され、後段のコンバータ212に供給されるとともに、必要に応じて、スイッチ303を介してRAM304に再び供給され、記憶される。

【0162】

符号長変換器305は、入力されるシンタックスと符号化パラメータとから、その符号化パラメータの符号長を決定し、アドレス発生回路306に出力する。アドレス発生回路306は、入力された符号長に対応して、上述したシフト量、ビットセレクト信号、書き込みアドレス、または読み出しアドレスを生成し、それらを、それぞれバレルシフタ302、スイッチ303、またはRAM304に供給する。

【0163】

以上のように、ヒストリVLC211は、いわゆる可変長符号化器として構成され、入力された符号化パラメータを可変長符号化して出力する。

【0164】

図21は、以上のようにしてヒストリフォーマット化されたデータをデコードするヒストリVLD203の構成例を表している。このヒストリVLD203には、コンバータ202から供給された符号化パラメータのデータがRAM311に供給されて、記憶される。このときの書き込みアドレスは、アドレス発生回路315から供給される。アドレス発生回路315はまた、所定のタイミングで読み出しアドレスを発生し、RAM311に供給する。このとき、RAM311は、読み出しアドレスに記憶されているデータを読み出し、バレルシフタ312に出力する。バレルシフタ312は、アドレス発生回路315が出力するシフト量に対応する分だけ、入力されるデータをシフトし、逆符号長変換器313と逆符号語変換器314に出力する。

【0165】

逆符号長変換器313にはまた、コンバータ202から、符号化パラメータが配置されているストリームのシンタックスの名称（項目NO.）が供給されている。逆符号長変換器313は、そのシンタックスに基づいて、入力されたデータ（符号語）から符号長を求め、求めた符号長をアドレス発生回路315に出力する。

【0166】

また、逆符号語変換器314は、バレルシフタ312より供給されたデータを、シンタックスに基づいて復号し（逆符号語化し）、ヒストリ情報多重化装置103に出力する。

【0167】

10

20

30

40

50

また、逆符号語変換器 3 1 4 は、どのような符号語が含まれているのかを特定するのに必要な情報（符号語の区切りを決定するのに必要な情報）を抽出し、アドレス発生回路 3 1 5 に出力する。アドレス発生回路 3 1 5 は、この情報と逆符号長変換器 3 1 3 より入力された符号長に基づいて、書き込みアドレスおよび読み出しアドレスを発生し、RAM 3 1 1 に出力するとともに、シフト量を発生し、バレルシフタ 3 1 2 に出力する。

【0 1 6 8】

図 2 2 は、コンバータ 2 1 2 の構成例を表している。この例においては、ヒストリ VLC 2 1 1 とコンバータ 2 1 2 の間に配置されているバッファメモリ 3 2 0 の、コントローラ 3 2 6 が出力する読み出しアドレスから 8 ビットのデータが読み出され、D 型フリップフロップ（D－F F）3 2 1 に供給され、保持されるようになされている。そして、D 型フリップフロップ 3 2 1 より読み出されたデータは、スタッフ回路 3 2 3 に供給されるとともに、8 ビットの D 型フリップフロップ 3 2 2 にも供給され、保持される。D 型フリップフロップ 3 2 2 より読み出された 8 ビットのデータは、D 型フリップフロップ 3 2 1 より読み出された 8 ビットのデータと合成され、1 6 ビットのパラレルデータとして、スタッフ回路 3 2 3 に供給される。

【0 1 6 9】

スタッフ回路 3 2 3 は、コントローラ 3 2 6 より供給されるスタッフ位置を示す信号（stuff position）の位置に符号 ” 1 ” を挿入し（スタッフイングし）、合計 1 7 ビットのデータとして、バレルシフタ 3 2 4 に出力する。

【0 1 7 0】

バレルシフタ 3 2 4 は、コントローラ 3 2 6 より供給されるシフト量を示す信号（shift）に基づいて入力されたデータをシフトして、8 ビットのデータを抽出し、8 ビットの D 型フリップフロップ 3 2 5 に出力する。D 型フリップフロップ 3 2 5 に保持されたデータは、そこから読み出され、バッファメモリ 3 2 7 を介して、後段のユーザデータフォーマッタ 2 1 3 に供給される。この時、コントローラ 3 2 6 は、出力するデータとともに、書き込みアドレスを発生し、コンバータ 2 1 2 とユーザデータフォーマッタ 2 1 3 との間に介在するバッファメモリ 3 2 7 に供給する。

【0 1 7 1】

図 2 3 は、スタッフ回路 3 2 3 の構成例を表している。D 型フリップフロップ 3 2 2、3 2 1 より入力された 1 6 ビットのデータは、それぞれスイッチ 3 3 1－1 6 乃至 3 3 1－1 の接点 a に入力されている。スイッチ 3 3 1－i（i＝0 乃至 1 5）の接点 c には、MSB 側（図中上方）に隣接するスイッチのデータが供給されている。例えば、スイッチ 3 3 1－1 2 の接点 c には、MSB 側に隣接するスイッチ 3 3 1－1 3 の接点 a に供給されている LSB から 1 3 番目のデータが供給されており、スイッチ 3 3 1－1 3 の接点 c には、MSB 側に隣接するスイッチ 3 3 1－1 4 の接点 a に供給されている LSB 側から 1 4 番目のデータが供給されている。

【0 1 7 2】

但し、LSB に対応するスイッチ 3 3 1－1 よりさらに下側のスイッチ 3 3 1－0 の接点 a は、開放されている。また、MSB に対応するスイッチ 3 3 1－1 6 の接点 c は、それより上位のスイッチが存在しないため、開放されている。

【0 1 7 3】

各スイッチ 3 3 1－0 乃至 3 3 1－1 6 の接点 b には、データ ” 1 ” が供給されている。

【0 1 7 4】

デコーダ 3 3 2 は、コントローラ 3 2 6 より供給されるデータ ” 1 ” を挿入する位置を示す信号 stuff position に対応して、スイッチ 3 3 1－0 乃至 3 3 1－1 6 のうち、1 つのスイッチを接点 b 側に切り替え、それより LSB 側のスイッチは、接点 c 側にそれぞれ切り替えさせ、それより MSB 側のスイッチは、接点 a 側に切り替えさせる。

【0 1 7 5】

図 2 3 は、LSB 側から 1 3 番目にデータ ” 1 ” を挿入する場合の例を示している。従っ

10

20

30

40

50

て、この場合、スイッチ 3 3 1-0 乃至スイッチ 3 3 1-1 2 は、いずれも接点 c 側に切り替えられ、スイッチ 3 3 1-1 3 は、接点 b 側に切り替えられ、スイッチ 3 3 1-1 4 乃至スイッチ 3 3 1-1 6 は、接点 a 側に切り替えられている。

【0176】

図 2 2 のコンバータ 2 1 2 は、以上のような構成により、2 2 ビットの符号を 2 3 ビットに変換して、出力することになる。

【0177】

図 2 4 は、図 2 2 のコンバータ 2 1 2 の各部の出力データのタイミングを表している。コンバータ 2 1 2 のコントローラ 3 2 6 がバイト単位のクロックに同期して、読み出しアドレス（図 2 4 (A)）を発生すると、バッファメモリ 3 2 0 から、それに対応するデータが、バイト単位で読み出され、D 型フリップフロップ 3 2 1 に一旦保持される。そして、D 型フリップフロップ 3 2 1 より読み出されたデータ（図 2 4 (B)）は、スタッフ回路 3 2 3 に供給されるとともに、D 型フリップフロップ 3 2 2 に供給され、保持される。D 型フリップフロップ 3 2 2 に保持されたデータは、そこからさらに読み出され（図 2 4 (C)）、スタッフ回路 3 2 3 に供給される。

10

【0178】

従って、スタッフ回路 3 2 3 の入力（図 2 4 (D)）は、読み出しアドレス A 1 のタイミングにおいて、最初の 1 バイトのデータ D 0 とされ、次の読み出しアドレス A 2 のタイミングにおいて、1 バイトのデータ D 0 と 1 バイトのデータ D 1 より構成される 2 バイトのデータとなり、さらに読み出しアドレス A 3 のタイミングにおいては、データ D 1 とデータ D 2 より構成される 2 バイトのデータとなる。

20

【0179】

スタッフ回路 3 2 3 には、データ " 1 " を挿入する位置を示す信号 stuff position（図 2 4 (E)）がコントローラ 3 2 6 より供給される。スタッフ回路 3 2 3 のデコーダ 3 3 2 は、スイッチ 3 3 1-1 6 乃至 3 3 1-0 のうち、この信号 stuff position に対応するスイッチを接点 b に切り換え、それより LSB 側のスイッチを接点 c 側に切り換え、さらにそれより MSB 側のスイッチを接点 a 側に切り換える。これにより、データ " 1 " が挿入されるので、スタッフ回路 3 2 3 からは、信号 stuff position で示す位置に、データ " 1 " が挿入されたデータ（図 2 4 (F)）が出力される。

【0180】

30

バレルシフタ 3 2 4 は、入力されたデータを、コントローラ 3 2 6 より供給される信号 shift（図 2 4 (G)）で示される量だけバレルシフトして、出力する（図 2 4 (H)）。この出力がさらに D 型フリップフロップ 3 2 5 で一旦保持された後、後段に出力される（図 2 4 (I)）。

【0181】

D 型フリップフロップ 3 2 5 より出力されるデータには、2 2 ビットのデータの次に、データ " 1 " が挿入されている。従って、データ " 1 " と、次のデータ " 1 " の間には、その間のビットが全て 0 であったとしても、0 のデータの連続する数は 2 2 となる。

【0182】

図 2 5 は、コンバータ 2 0 2 の構成例を表している。このコンバータ 2 0 2 の D 型フリップフロップ 3 4 1 乃至コントローラ 3 4 6 よりなる構成は、図 2 2 に示したコンバータ 2 1 2 の D 型フリップフロップ 3 2 1 乃至コントローラ 3 2 6 と基本的に同様の構成であるが、コンバータ 2 1 2 におけるスタッフ回路 3 2 3 に代えて、ディリット回路 3 4 3 が挿入されている点がコンバータ 2 1 2 における場合と異なっている。その他の構成は、図 2 2 のコンバータ 2 1 2 における場合と同様である。

40

【0183】

すなわち、このコンバータ 2 0 2 においては、コントローラ 3 4 6 が出力する削除するビットの位置を示す信号 delete position に従って、ディリット回路 3 4 3 が、そのビット（図 2 2 のスタッフ回路 3 2 3 で挿入されたデータ " 1 "）が削除される。

【0184】

50

その他の動作は、図 2 2 のコンバータ 2 1 2 における場合と同様である。

【0185】

図 2 6 は、ディリート回路 3 4 3 の構成例を表している。この構成例においては、D 型フリップフロップ 3 4 2、3 4 1 より入力された 1 6 ビットのデータのうち、LSB 側の 1 5 ビットが、それぞれ対応するスイッチ 3 5 1-0 乃至 3 5 1-1 4 の接点 a に供給されている。各スイッチの接点 b には、1 ビットだけ MSB 側のデータが供給されている。デコーダ 3 5 2 は、コントローラ 3 4 6 より供給される信号 delete position により指定されるビットを削除して、1 5 ビットのデータとして出力するようになされている。

【0186】

図 2 6 は、LSB から第 1 3 番目のビットがディリートされる状態を示している。従って、この場合、スイッチ 3 5 1-0 乃至スイッチ 3 5 1-1 1 が接点 a 側に切り替えられ、LSB から第 1 2 番目までの 1 2 ビットが、そのまま選択、出力されている。また、スイッチ 3 5 1-1 2 乃至 3 5 1-1 4 は、それぞれ接点 b 側に切り替えられているので、第 1 4 番目乃至第 1 6 番目のデータが、第 1 3 番目乃至第 1 5 番目のビットのデータとして選択、出力される。

【0187】

図 2 3 のスタッフ回路 3 2 3 および図 2 6 のディリート回路 3 4 3 の入力が 1 6 ビットとなっているのは、それぞれ図 2 2 のコンバータ 2 1 2 のスタッフ回路 3 2 3 の入力が、D 型フリップフロップ 3 2 2、3 2 1 より供給される 1 6 ビットとされており、また、図 2 5 のコンバータ 2 0 2 においても、ディリート回路 3 4 3 の入力が、D 型フリップフロップ 3 4 2、3 4 1 により 1 6 ビットとされているためである。図 2 2 において、スタッフ回路 3 2 3 の出力する 1 7 ビットをバレルシフタ 3 2 4 でバレルシフトすることにより、例えば 8 ビットを最終的に選択、出力しているのと同様に、図 2 5 のコンバータ 2 0 2 においても、ディリート回路 3 4 3 の出力する 1 5 ビットのデータを、バレルシフタ 3 4 4 で所定量だけバレルシフトすることにより、8 ビットのデータとしている。

【0188】

図 2 7 は、コンバータ 2 1 2 の他の構成例を表している。この構成例においては、カウンタ 3 6 1 が入力データのうち、連続する 0 のビットの数をカウントし、そのカウント結果をコントローラ 3 2 6 に出力するようになされている。コントローラ 3 2 6 は、例えばカウンタ 3 6 1 が連続する 0 のビットを 2 2 個カウントしたとき、信号 stuff position をスタッフ回路 3 2 3 に出力する。また、このとき、コントローラ 3 2 6 は、カウンタ 3 6 1 をリセットし、再び連続する 0 のビットの数をカウンタ 3 6 1 にカウントさせる。

【0189】

その他の構成と動作は、図 2 2 における場合と同様である。

【0190】

図 2 8 は、コンバータ 2 0 2 の他の構成例を表している。この構成例においては、入力データのうち、連続する 0 の数をカウンタ 3 7 1 がカウントし、そのカウント結果をコントローラ 3 4 6 に出力するようになされている。カウンタ 3 7 1 のカウント値が 2 2 に達したとき、コントローラ 3 4 6 は、信号 delete position をディリート回路 3 4 3 に出力するとともに、カウンタ 3 7 1 をリセットし、再び新たな連続する 0 のビットの数をカウンタ 3 7 1 にカウントさせる。その他の構成は、図 2 5 における場合と同様である。

【0191】

このように、この構成例においては、所定のパターン（データ " 0 " の連続する数）に基づいて、マーカービットとしてのデータ " 1 " が挿入され、また、削除されることになる。

【0192】

図 2 7 と図 2 8 に示す構成は、図 2 2 と図 2 5 に示す構成よりも効率的な処理が可能となる。但し、変換後の長さが元の履歴情報に依存することになる。

【0193】

図 2 9 は、ユーザデータフォーマッタ 2 1 3 の構成例を表している。この例においては

、コントローラ 3 8 3 がコンバータ 2 1 2 とユーザデータフォーマッタ 2 1 3 との間に配置されているバッファメモリ（図示せず）に読み出しアドレスを出力すると、そこから読み出されたデータが、ユーザデータフォーマッタ 2 1 3 のスイッチ 3 8 2 の接点 a 側に供給される。ROM 3 8 1 には、ユーザデータスタートコード、データ I D などの user_data() を生成するのに必要なデータが記憶されている。コントローラ 3 1 3 は、所定のタイミングにおいて、スイッチ 3 8 2 を接点 a 側または接点 b 側に切り替え、ROM 3 8 1 に記憶されているデータ、またはコンバータ 2 1 2 より供給されるデータを適宜選択し、出力する。これにより、user_data() のフォーマットのデータが符号化装置 1 0 6 に出力される。

【0194】

なお、図示は省略するが、ユーザデータデコーダ 2 0 1 は、図 2 9 の ROM 3 8 1 より読み出され、挿入されたデータを削除するスイッチを介して、入力データを出力するようにすることで実現することができる。

10

【0195】

図 3 0 は、例えば映像編集スタジオにおいて、複数のトランスコーダ 1 0 1 - 1 乃至 1 0 1 - N が直列に接続されて使用される状態を示している。各トランスコーダ 1 0 1 - i ($i = 1$ 乃至 N) の履歴情報多重化装置 1 0 3 - i は、上述した符号化パラメータ用の領域の最も古い符号化パラメータが記録されている区画に、自己が用いた最新の符号化パラメータを上書きする。このことにより、ベースバンドの画像データには、同一のマクロブロックに対応する直近の 4 世代分の符号化パラメータ（世代履歴情報）が記録されることになる（図 1 8）。

20

【0196】

各符号化装置 1 0 6 - i の符号化装置 1 0 6 - i（図 1 9）は、その可変長符号化回路 5 8 において、履歴情報分離装置 1 0 5 - i から供給される今回用いる符号化パラメータに基づいて、量子化回路 5 7 より供給されるビデオデータを符号化する。このようにして生成されるビットストリーム（例えば、picture_header()）中に、その現符号化パラメータは多重化される。

【0197】

可変長符号化回路 5 8 はまた、履歴エンコーディング装置 1 0 7 - i より供給されるユーザデータ（世代履歴情報を含む）を、出力するビットストリーム中に多重化する（図 1 8 に示すような埋め込み処理ではなく、ビットストリーム中に多重化する）。そして、符号化装置 1 0 6 - i の出力するビットストリームは、SDTI (Serial Data Transfer Interface) 1 0 8 - i を介して、後段のトランスコーダ 1 0 1 - ($i + 1$) に入力される。

30

【0198】

トランスコーダ 1 0 1 - i とトランスコーダ 1 0 1 - ($i + 1$) は、それぞれ図 1 5 に示すように構成されている。従って、その処理は、図 1 5 を参照して説明した場合と同様となる。

【0199】

実際の符号化パラメータの履歴を利用した符号化として、現在 I ピクチャとして符号化されていたものを、P もしくは B ピクチャに変更したい場合、過去の符号化パラメータの履歴を見て、過去に P もしくは B ピクチャであった場合を探し、これらの履歴が存在した場合は、その動きベクトルなどのパラメータを利用して、ピクチャタイプを変更する。反対に過去に履歴がない場合は、動き検出を行わないピクチャタイプの変更を断念する。もちろん履歴がない場合であっても、動き検出を行えばピクチャタイプを変更できる。

40

【0200】

図 1 8 に示すフォーマットの場合、4 世代分の符号化パラメータを埋め込むようにしたが、I、P、B の各ピクチャタイプのパラメータを埋め込むようにすることもできる。図 3 1 は、この場合のフォーマットの例を示している。この例では、同一のマクロブロックが、過去にピクチャタイプの変更を伴って符号化されたときにおける、ピクチャタイプ毎に 1 世代分の符号化パラメータ（ピクチャ履歴情報）が記録される。したがって、図 1 6 に示した復号化装置 1 0 2、および図 1 9 に示した符号化装置 1 0 6 は、現在（最新）、

50

第3世代、第2世代、および第1世代の符号化パラメータの代わりに、Iピクチャ、Pピクチャ、およびBピクチャに対応する1世代分の符号化パラメータを入出力することになる。

【0201】

また、この例の場合、Cb[1][x]とCr[1][x]の空き領域は利用しないので、Cb[1][x]とCr[1][x]の領域を有さない4:2:0フォーマットの画像データにも本発明を適用することができる。

【0202】

この例の場合、復号装置102は、符号化パラメータを復号と同時に取り出し、ピクチャタイプを判定して、画像信号のピクチャタイプに対応した場所に符号化パラメータを書き込んで（多重化して）履歴情報分離装置105に出力する。履歴情報分離装置105は、符号化パラメータを分離し、これから符号化したいピクチャタイプと、入力された過去の符号化パラメータを考慮して、ピクチャタイプを変更しながら再符号化を行うことができる。

10

【0203】

次に、各トランスコーダ101において、変更が可能なピクチャタイプを判定する処理について、図32のフローチャートを参照して説明する。なお、トランスコーダ101におけるピクチャタイプの変更は、過去の動きベクトルを利用するので、この処理は動き検出を行わないで実行されることを前提としている。また、以下に説明する処理は、履歴情報分離装置105により実行される。

20

【0204】

ステップS1において、ピクチャタイプ毎に1世代分の符号化パラメータ（ピクチャ履歴情報）が履歴情報分離装置105に入力される。

【0205】

ステップS2において、履歴情報分離装置105は、ピクチャ履歴情報の中に、Bピクチャに変更したときの符号化パラメータが存在するか否かを判定する。ピクチャ履歴情報にBピクチャに変更したときの符号化パラメータが存在すると判定された場合、ステップS3に進む。

【0206】

ステップS3において、履歴情報分離装置105は、ピクチャ履歴情報の中に、Pピクチャに変更したときの符号化パラメータが存在するか否かを判定する。ピクチャ履歴情報にPピクチャに変更したときの符号化パラメータが存在すると判定された場合、ステップS4に進む。

30

【0207】

ステップS4において、履歴情報分離装置105は、変更可能なピクチャタイプがIピクチャ、Pピクチャ、およびBピクチャであると判断する。

【0208】

ステップS3において、ピクチャ履歴情報にPピクチャに変更したときの符号化パラメータが存在しないと判定された場合、ステップS5に進む。

【0209】

ステップS5において、履歴情報分離装置105は、変更可能なピクチャタイプがIピクチャ、およびBピクチャであると判断する。さらに、履歴情報分離装置105は、特殊処理（Bピクチャの履歴情報に含まれる後方予測ベクトルを使わず、前方予測ベクトルだけを使う）を施すことにより、擬似的にPピクチャに変更可能であると判断する。

40

【0210】

ステップS2において、ピクチャ履歴情報にBピクチャに変更したときの符号化パラメータが存在しないと判定された場合、ステップS6に進む。

【0211】

ステップS6において、履歴情報分離装置105は、ピクチャ履歴情報にPピクチャ

50

ャに変更したときの符号化パラメータが存在するか否かを判定する。ピクチャ履歴情報に P ピクチャに変更したときの符号化パラメータが存在すると判定された場合、ステップ S 7 に進む。

【0212】

ステップ S 7 において、ヒストリ情報分離装置 105 は、変更可能なピクチャタイプが I ピクチャ、および P ピクチャであると判断する。さらに、ヒストリ情報分離装置 105 は、特殊処理（P ピクチャに履歴情報に含まれる前方予測ベクトルだけを使う）を施すことにより、B ピクチャに変更可能であると判断する。

【0213】

ステップ S 6 において、ピクチャ履歴情報に P ピクチャに変更したときの符号化パラメータが存在しないと判定された場合、ステップ S 8 に進む。ステップ S 8 において、ヒストリ情報分離装置 105 は、動きベクトルが存在しないので、変更可能なピクチャタイプが I ピクチャだけである（I ピクチャなので I ピクチャ以外には変更できない）と判断する。

【0214】

ステップ S 4, S 5, S 7, S 8 の処理の次にステップ S 9 において、ヒストリ情報分離装置 105 は、変更可能なピクチャタイプを表示装置（図示せず）に表示してユーザに通知する。

【0215】

図 33 は、ピクチャタイプ変更の例を示している。ピクチャタイプを変更する場合、GOP を構成するフレーム数を変更される。すなわち、この例の場合、 $N=15$ （GOP のフレーム数 $N=15$ ）、 $M=3$ （GOP 内の I、または P ピクチャの出現周期 $M=3$ ）のフレームから構成される 4 Mbps の Long GOP（第 1 世代）から、 $N=1$, $M=1$ のフレームで構成される 50 Mbps の Short GOP（第 2 世代）に変換され、再度、 $N=15$, $M=3$ のフレームから構成される 4 Mbps の Long GOP（第 3 世代）に変換されている。なお、図中において破線は、GOP の境界を示している。

【0216】

第 1 世代から第 2 世代にピクチャタイプが変更される場合において、上述した変更可能ピクチャタイプ判定処理の説明から明らかなように、全てのフレームは、ピクチャタイプを I ピクチャに変更することが可能である。このピクチャタイプ変更のとき、動画像（第 0 世代）が第 1 世代に変換されたときに演算された全ての動きベクトルは、ピクチャ履歴情報に保存された（残された）状態となる。次に、再度 Long GOP に変換される（第 2 世代から第 3 世代にピクチャタイプが変更される）場合、第 0 世代から第 1 世代に変換されたときのピクチャタイプ毎の動きベクトルが保存されているので、これを再利用することにより、画質劣化を抑えて、再度、Long GOP に変換することが可能となる。

【0217】

図 34 は、ピクチャタイプ変更の他の例を示している。この例の場合、 $N=14$, $M=2$ である 4 Mbps の Long GOP（第 1 世代）から、 $N=2$, $M=2$ である 18 Mbps の Short GOP（第 2 世代）に変換され、さらに、 $N=1$, $M=1$ であるフレーム数が 1 の 50 Mbps の Short GOP（第 3 世代）に変換されて、1 Mbps の、フレーム数 N がランダムな GOP（第 4 世代）に変換される。

【0218】

この例においても、第 0 世代から第 1 世代に変換されたときのピクチャタイプ毎の動きベクトルが、第 3 世代から第 4 世代への変換のときまで保存される。そこで、図 34 に示すように、複雑にピクチャタイプを変更しても、保存されている符号化パラメータを再利用されることにより、画質劣化を小さく抑えることができる。さらに、保存されている符号化パラメータの量子化スケールを有効に利用すれば画質劣化の少ない符号化を実現できる。

【0219】

この量子化スケールの再利用について、図 35 を参照して説明する。図 35 は、所定のフレームが、第 1 世代から第 4 世代まで常に、I ピクチャに変換されており、ビットレートだけが、4 Mbps, 18 Mbps, または 50 Mbps に変更されていることを示している。

【0220】

例えば、第1世代(4Mbps)から第2世代(18Mbps)への変換の際に、ビットレートの高速化に伴って、細かい量子化スケールで再符号化しても画質は向上しない。なぜならば、過去において粗い量子化ステップで量子化されたデータは、復元しないからである。したがって、図35に示すように、途中でビットレートが高速化しても、それに伴って細かい量子化ステップで量子化することは、情報量が増加するだけであって画質の向上には繋がらない。したがって、過去のもっとも粗い(大きい)量子化スケールを維持するように制御すれば、最も無駄が無く、効率的な符号化が可能となる。

【0221】

なお、第3世代から第4世代への変更時には、ビットレートは、50Mbpsから4Mbpsに低下されているが、この場合にも、過去のもっとも粗い(大きい)量子化スケールが維持される。

10

【0222】

上述したように、ビットレートが変更されるときは、過去の量子化スケールの履歴を利用して符号化することは非常に有効である。

【0223】

この量子化制御処理について、図36のフローチャートを参照して説明する。ステップS11において、ヒストリ情報分離装置105は、入力されたピクチャ履歴情報に、いまから変換するピクチャタイプの符号化パラメータが存在するか否かを判定する。変換するピクチャタイプの符号化パラメータが存在すると判定された場合、ステップS12に進む。

20

【0224】

ステップS12において、ヒストリ情報分離装置105は、ピクチャ履歴情報の対象となる符号化パラメータから、history_q_scale_codeを抽出する。

【0225】

ステップS13において、ヒストリ情報分離装置105は、送信バッファ59から量子化回路57にフィードバックされるバッファ残量に基づいて、feedback_q_scale_codeを演算する。

【0226】

ステップS14において、ヒストリ情報分離装置105は、history_q_scale_codeがfeedback_q_scale_codeよりも大きい(粗い)か否かを判定する。history_q_scale_codeがfeedback_q_scale_codeよりも大きいと判定された場合、ステップS15に進む。

30

【0227】

ステップS15において、ヒストリ情報分離装置105は、量子化スケールとしてhistory_q_scale_codeを量子化回路57に出力する。量子化回路57は、history_q_scale_codeを用いて量子化を実行する。

【0228】

ステップS16において、フレームに含まれる全てのマクロブロックが量子化されたか否かが判定される。全てのマクロブロックがまだ量子化されていないと判定された場合、ステップS12に戻り、ステップS12乃至S16の処理が、全てのマクロブロックが量子化されるまで繰り返される。

40

【0229】

ステップS14において、history_q_scale_codeがfeedback_q_scale_codeよりも大きくない(細かい)いと判定された場合、ステップS17に進む。

【0230】

ステップS17において、ヒストリ情報分離装置105は、量子化スケールとしてfeedback_q_scale_codeを量子化回路57に出力する。量子化回路57は、feedback_q_scale_codeを用いて量子化を実行する。

【0231】

ステップS11において、変換するピクチャタイプの符号化パラメータが、ヒストリ情

50

報中に存在しないと判定された場合、ステップ S 1 8 に進む。

【0232】

ステップ S 1 8 において、ヒストリ情報分離装置 1 0 5 は、送信バッファ 5 9 から量子化回路 5 7 にフィードバックされるバッファ残量に基づいて、feedback_q_scale_code を演算する。

【0233】

ステップ S 1 9 において、量子化回路 5 7 は、Feedback_q_scale_code を用いて量子化を実行する。

【0234】

ステップ S 2 0 において、フレームに含まれる全てのマクロブロックが量子化されたか否かが判定される。全てのマクロブロックがまだ量子化されていないと判定された場合、ステップ S 1 8 に戻り、ステップ S 1 8 乃至 S 2 0 の処理が、全てのマクロブロックが量子化されるまで繰り返される。

10

【0235】

なお、本実施の形態におけるトランスコーダ 1 0 1 の内部においては、上述したように、復号側と符号側が粗結合されており、符号化パラメータを画像データに多重化させて伝送させたが、図 3 7 に示すように、復号装置 1 0 2 と符号化装置 1 0 6 を直接接続する（密結合する）ようにしてもよい。

【0236】

図 1 5 において説明したトランスコーダ 1 0 1 は、第 1 世代から第 3 世代の過去の符号化パラメータを符号化装置 1 0 6 に供給するために、ベースバンドビデオデータに過去の符号化パラメータを多重化して伝送するようにしていた。しかしながら、本発明においては、ベースバンドビデオデータに過去の符号化パラメータを多重化する技術は必須ではなく、図 3 7 に示されたように、ベースバンドビデオデータとは異なる伝送路（たとえばデータ転送パス）を使用して、過去の符号化パラメータを伝送するようにしても良い。

20

【0237】

つまり、図 3 7 に示した、復号装置 1 0 2、ヒストリデコーディング装置 1 0 4、符号化装置 1 0 6 及びヒストリエンコーディング装置 1 0 7 は、図 1 5 において説明した復号装置 1 0 2、ヒストリデコーディング装置 1 0 4、符号化装置 1 0 6 及びヒストリエンコーディング装置 1 0 7 とまったく同じ機能及び構成を有している。

30

【0238】

復号装置 1 0 2 の可変長復号回路 1 1 2 は、第 3 世代の符号化ストリーム S T (3rd) のシーケンス層、GOP 層、ピクチャ層、スライス層及びマクロブロック層から、第 3 世代の符号化パラメータを抽出し、それを、ヒストリエンコーディング装置 1 0 7 及び符号化装置 1 0 6 のコントローラ 7 0 にそれぞれ供給する。ヒストリエンコーディング装置 1 0 7 は、受け取った第 3 世代の符号化パラメータをピクチャ層のユーザデータエリアに記述できるように converted_history_stream() に変換し、converted_history_stream() をユーザデータとして符号化装置 1 0 6 の可変長符号化回路 5 8 に供給する。

【0239】

さらに可変長復号回路 1 1 2 は、第 3 世代の符号化ストリームのピクチャ層のユーザデータエリアから、第 1 世代の符号化パラメータ及び第 2 世代の符号化パラメータを含んでいるユーザデータ user_data を抽出し、ヒストリデコーディング装置 1 0 4 及び符号化装置 1 0 6 の可変長符号化回路 5 8 に供給する。ヒストリデコーディング装置 1 0 4 は、ユーザデータエリアに converted_history_stream() として記述されたヒストリストリームから、第 1 世代の符号化パラメータ及び第 2 世代の符号化パラメータを抽出し、それを符号化装置 1 0 6 のコントローラに供給する。

40

【0240】

符号化装置 1 0 6 のコントローラ 7 0 は、ヒストリデコーディング装置 1 0 4 から受け取った第 1 世代及び第 2 世代の符号化パラメータと、符号化装置 1 0 2 から受け取った第 3 世代の符号化パラメータとに基づいて、符号化装置 1 0 6 の符号化処理をコントロール

50

する。

【0241】

符号化装置106の可変長符号化回路58は、復号装置102から第1世代の符号化パラメータ及び第2の符号化パラメータを含んでいるユーザデータuser_dataを受け取るとともに、ヒストリエンコーディング装置107から第3世代の符号化パラメータを含んでいるユーザデータuser_dataを受け取り、それらのユーザデータをヒストリ情報として、第4世代の符号化ストリームのピクチャ層のユーザデータエリアに記述する。

【0242】

図38は、MPEGのビデオストリームをデコードするためのシンタックスを表わした図である。デコーダは、このシンタックスに従ってMPEGビットストリームをデコードすることによって、ビットストリームから意味のある複数のデータ項目（データエレメント）を抽出する。以下に説明するシンタックスは、図において、その関数や条件文は細活字で表わされ、そのデータエレメントは、太活字で表されている。データ項目は、その名称、ビット長、及びそのタイプと伝送順序を示すニーモニック（Mnemonic）で記述されている。

【0243】

まず、この図38に示されているシンタックスにおいて使用されている関数について説明する。

【0244】

next_start_code()関数は、ビットストリーム中に記述されているスタートコードを探すための関数である。この図38に示されたシンタックスにおいて、このnext_start_code()関数の次に、sequence_header()関数とsequence_extension()関数とが順に配置されているので、このビットストリームには、このsequence_header()関数とsequence_extension()関数によって定義されたデータエレメントが記述されている。従って、ビットストリームのデコード時には、このnext_start_code()関数によって、sequence_header()関数とsequence_extension()関数の先頭に記述されているスタートコード（データエレメントの一種）をビットストリーム中から見つけ、それを基準にして、sequence_header()関数とsequence_extension()関数をさらに見つけ、それらによって定義された各データエレメントをデコードする。

【0245】

尚、sequence_header()関数は、MPEGビットストリームのシーケンス層のヘッダデータを定義するための関数であって、sequence_extension()関数は、MPEGビットストリームのシーケンス層の拡張データを定義するための関数である。

【0246】

sequence_extension()関数の次に配置されている do{ }while構文は、while文によって定義されている条件が真である間、do文の{ }内の関数に基いて記述されたデータエレメントをデータストリーム中から抽出するための構文である。すなわち、do{ }while構文によって、while文によって定義されている条件が真である間、ビットストリーム中から、do文内の関数に基いて記述されたデータエレメントを抽出するデコード処理が行われる。

【0247】

このwhile文に使用されているnextbits()関数は、ビットストリーム中に現れるビット又はビット列と、次にデコードされるデータエレメントとを比較するための関数である。この図38のシンタックスの例では、nextbits()関数は、ビットストリーム中のビット列とビデオシーケンスの終わりを示すsequence_end_codeとを比較し、ビットストリーム中のビット列とsequence_end_codeとが一致しないときに、このwhile文の条件が真となる。従って、sequence_extension()関数の次に配置されている do{ }while構文は、ビットストリーム中に、ビデオシーケンスの終わりを示すsequence_end_codeが現れない間、do文中の関数によって定義されたデータエレメントがビットストリーム中に記述されていることを示している。

【0248】

ビットストリーム中には、sequence_extension()関数によって定義された各データエレ

10

20

30

40

50

メントの次には、extension_and_user_data(0)関数によって定義されたデータエレメントが記述されている。このextension_and_user_data(0)関数は、MPEGビットストリームのシーケンス層の拡張データとユーザデータを定義するための関数である。

【0249】

このextension_and_user_data(0)関数の次に配置されている do{ }while構文は、while文によって定義されている条件が真である間、do文の{ }内の関数に基いて記述されたデータエレメントを、ビットストリーム中から抽出するための関数である。このwhile文において使用されているnextbits()関数は、ビットストリーム中に現れるビット又はビット列と、picture_start_code又はgroup_start_codeとの一致を判断するための関数であって、ビットストリーム中に現れるビット又はビット列と、picture_start_code又はgroup_start_codeとが一致する場合には、while文によって定義された条件が真となる。よって、このdo{ }while構文は、ビットストリーム中において、picture_start_code又はgroup_start_codeが現れた場合には、そのスタートコードの次に、do文中の関数によって定義されたデータエレメントのコードが記述されているので、このpicture_start_code又はgroup_start_codeによって示されるスタートコードを探し出すことによって、ビットストリーム中からdo文中に定義されたデータエレメントを抽出することができる。

10

【0250】

このdo文の最初に記述されているif文は、ビットストリーム中にgroup_start_codeが現れた場合、という条件を示している。このif文による条件が真である場合には、ビットストリーム中には、このgroup_start_codeの次にgroup_of_picture_header(1)関数及びextension_and_user_data(1)関数によって定義されているデータエレメントが順に記述されている。

20

【0251】

このgroup_of_picture_header(1)関数は、MPEGビットストリームのGOP層のヘッダデータを定義するための関数であって、extension_and_user_data(1)関数は、MPEGビットストリームのGOP層の拡張データ(extension_data)及びユーザデータ(user_data)を定義するための関数である。

【0252】

さらに、このビットストリーム中には、group_of_picture_header(1)関数及びextension_and_user_data(1)関数によって定義されているデータエレメントの次に、picture_header()関数とpicture_coding_extension()関数によって定義されたデータエレメントが記述されている。もちろん、先に説明したif文の条件が真とならない場合には、group_of_picture_header(1)関数及びextension_and_user_data(1)関数によって定義されているデータエレメントは記述されていないので、extension_and_user_data(0)関数によって定義されているデータエレメントの次に、picture_header()関数とpicture_coding_extension()関数によって定義されたデータエレメントが記述されている。

30

【0253】

このpicture_header()関数は、MPEGビットストリームのピクチャ層のヘッダデータを定義するための関数であって、picture_coding_extension()関数は、MPEGビットストリームのピクチャ層の第1の拡張データを定義するための関数である。

40

【0254】

次のwhile文は、このwhile文によって定義されている条件が真である間、次のif文の条件判断を行うための関数である。このwhile文において使用されているnextbits()関数は、ビットストリーム中に現れるビット列と、extension_start_code又はuser_data_start_codeとの一致を判断するための関数であって、ビットストリーム中に現れるビット列と、extension_start_code又はuser_data_start_codeとが一致する場合には、このwhile文によって定義された条件が真となる。

【0255】

第1のif文は、ビットストリーム中に現れるビット列とextension_start_codeとの一致を判断するための関数である。ビットストリーム中に現れるビット列と32ビットのexte

50

extension_start_codeとが一致する場合には、ビットストリーム中において、extension_start_codeの次にextension_data(2)関数によって定義されるデータエレメントが記述されている。

【0256】

第2のif文は、ビットストリーム中に現れるビット列とuser_data_start_codeとの一致を判断するための構文であって、ビットストリーム中に現れるビット列と32ビットのuser_data_start_codeとが一致する場合には、第3のif文の条件判断が行われる。このuser_data_start_codeは、MPEGビットストリームのピクチャ層のユーザデータエリアの開始を示すためのスタートコードである。

【0257】

10

第3のif文は、ビットストリーム中に現れるビット列とHistory_Data_IDとの一致を判断するための構文である。ビットストリーム中に現れるビット列とこの32ビットのHistory_Data_IDとが一致する場合には、このMPEGビットストリームのピクチャ層のユーザデータエリアにおいて、この32ビットのHistory_Data_IDによって示されるコードの次に、converted_history_stream()関数によって定義されるデータエレメントが記述されている。

【0258】

converted_history_stream()関数は、MPEG符号化時に使用したあらゆる符号化パラメータを伝送するための履歴情報及び履歴データを記述するための関数である。このconverted_history_stream()関数によって定義されているデータエレメントの詳細は、図40乃至図47を参照して、history_stream()として後述する。また、このHistory_Data_IDは、MPEGビットストリームのピクチャ層のユーザデータエリアに記述されたこの履歴情報及び履歴データが記述されている先頭を示すためのスタートコードである。

20

【0259】

else文は、第3のif文において、条件が非真であることを示すための構文である。従って、このMPEGビットストリームのピクチャ層のユーザデータエリアにおいて、converted_history_stream()関数によって定義されたデータエレメントが記述されていない場合には、user_data()関数によって定義されたデータエレメントが記述されている。

【0260】

図38において、履歴情報は、converted_history_stream()に記述され、user_data()に記述される訳ではないが、このconverted_history_stream()は、MPEG規格のuser_dataの一種として記述される。そこで、本明細書中においては、場合によって、履歴情報がuser_dataに記述されるとも説明するが、それは、MPEG規格のuser_dataの一種として記述されるということを意味する。

30

【0261】

picture_data()関数は、MPEGビットストリームのピクチャ層のユーザデータの次に、スライス層及びマクロブロック層に関するデータエレメントを記述するための関数である。通常は、このpicture_data()関数によって示されるデータエレメントは、ビットストリームのピクチャ層のユーザデータエリアに記述されたconverted_history_stream()関数によって定義されるデータエレメント又はuser_data()関数によって定義されたデータエレメントの次に記述されているが、ピクチャ層のデータエレメントを示すビットストリーム中に、extension_start_code又はuser_data_start_codeが存在しない場合には、このpicture_data()関数によって示されるデータエレメントは、picture_coding_extension()関数によって定義されるデータエレメントの次に記述されている。

40

【0262】

このpicture_data()関数によって示されるデータエレメントの次には、sequence_header()関数とsequence_extension()関数とによって定義されたデータエレメントが順に配置されている。このsequence_header()関数とsequence_extension()関数によって記述されたデータエレメントは、ビデオストリームのシーケンスの先頭に記述されたsequence_header()関数とsequence_extension()関数によって記述されたデータエレメントと全く同じ

50

である。このように同じデータをストリーム中に記述する理由は、ビットストリーム受信装置側でデータストリームの途中（例えばピクチャ層に対応するビットストリーム部分）から受信が開始された場合に、シーケンス層のデータを受信できなくなり、ストリームをデコード出来なくなることを防止するためである。

【0263】

この最後のsequence_header()関数とsequence_extension()関数とによって定義されたデータエレメントの次、つまり、データストリームの最後には、シーケンスの終わりを示す32ビットのsequence_end_codeが記述されている。

【0264】

以上のシンタックスの基本的な構成の概略を示すと、図39に示すようになる。

10

【0265】

次に、converted_history_stream()関数によって定義されたヒストリストリームに関して説明する。

【0266】

このconverted_history_stream()は、MPEGのピクチャ層のユーザデータエリアに履歴情報を示すヒストリストリームを挿入するための関数である。尚、「converted」の意味は、スタートエミュレーションを防止するために、ユーザエリアに挿入すべき履歴データから構成される履歴ストリームの少なくとも22ビット毎にマーカービット（1ビット）を挿入する変換処理を行ったストリームであることを意味している。

20

【0267】

このconverted_history_stream()は、以下に説明する固定長の履歴ストリーム（図40乃至図46）又は可変長の履歴ストリーム（図47）のいずれかの形式で記述される。エンコーダ側において固定長の履歴ストリームを選択した場合には、デコーダ側において履歴ストリームから各データエレメントをデコードするための回路及びソフトウェアが簡単になるというメリットがある。一方、エンコーダ側において可変長の履歴ストリームを選択した場合には、エンコーダにおいてピクチャ層のユーザエリアに記述される履歴情報（データエレメント）を必要に応じて任意に選択することができるので、履歴ストリームのデータ量を少なくすることができ、その結果、符号化されたビットストリーム全体のデータレートを低減することができる。

30

【0268】

本発明において説明する「履歴ストリーム」、「ヒストリストリーム」、「履歴情報」、「ヒストリ情報」、「履歴データ」、「ヒストリデータ」、「履歴パラメータ」、「ヒストリパラメータ」とは、過去の符号化処理において使用した符号化パラメータ（又はデータエレメント）を意味し、現在の（最終段の）符号化処理において使用した符号化パラメータを意味するものではない。例えば、第1世代の符号化処理において、あるピクチャをIピクチャで符号化して伝送し、次なる第2世代の符号化処理において、このピクチャを今度はPピクチャとして符号化して伝送し、さらに、第3世代の符号化処理において、このピクチャをBピクチャで符号化して伝送する例をあげて説明する。

【0269】

第3世代の符号化処理において使用した符号化パラメータが、第3世代の符号化処理において生成された符号化ビットストリームのシーケンス層、GOP層、ピクチャ層、スライス層及びマクロブロック層の所定位置に記述されている。一方、過去の符号化処理である第1世代及び第2世代の符号化処理において使用した符号化パラメータは、第3世代の符号化処理において使用した符号化パラメータが記述されるシーケンス層やGOP層に記述されるのでは無く、既に説明したシンタックスに従って、符号化パラメータの履歴情報として、ピクチャ層のユーザデータエリアに記述される。

40

【0270】

まず、固定長の履歴ストリームシンタックスについて図40乃至図46を参照して説明する。

【0271】

50

最終段（例えば第3世代）の符号化処理において生成されたビットストリームのピクチャ層のユーザデータエリアには、まず最初に、過去（例えば第1世代及び第2世代）の符号化処理において使用されていたシーケンス層のシーケンスヘッダに含められる符号化パラメータが、履歴ストリームとして挿入される。尚、過去の符号化処理において生成されたビットストリームのシーケンス層のシーケンスヘッダ等の履歴情報は、最終段の符号化処理において生成されたビットストリームのシーケンス層のシーケンスヘッダに挿入されることは無いという点に注意すべきである。

【0272】

過去の符号化処理で使用したシーケンスヘッダ（sequence_header）に含められるデータ要素は、sequence_header_code、sequence_header_present_flag、horizontal_size_value、marker_bit、vertical_size_value、aspect_ratio_information、frame_rate_code、bit_rate_value、VBV_buffer_size_value、constrained_parameter_flag、load_intra_quantiser_matrix、load_non_intra_quantiser_matrix、intra_quantiser_matrix、及びnon_intra_quantiser_matrix等から構成される。

【0273】

sequence_header_codeは、シーケンス層のスタート同期コードを表すデータである。sequence_header_present_flagは、sequence_header内のデータが有効か無効かを示すデータである。horizontal_size_valueは、画像の水平方向の画素数の下位12ビットから成るデータである。marker_bitは、スタートコードエミュレーションを防止するために挿入されるビットデータである。vertical_size_valueは、画像の縦のライン数の下位12ビットからなるデータである。aspect_ratio_informationは、画素のアスペクト比（縦横比）または表示画面アスペクト比を表すデータである。frame_rate_codeは、画像の表示周期を表すデータである。

【0274】

bit_rate_valueは、発生ビット量に対する制限のためのビット・レートの下位18ビット（400bsp単位で切り上げる）データである。VBV_buffer_size_valueは、発生符号量制御用の仮想バッファ（ビデオバッファベリファイヤー）の大きさを決める値の下位10ビットデータである。constrained_parameter_flagは、各パラメータが制限以内であることを示すデータである。load_intra_quantiser_matrixは、イントラMB用量子化マトリックス・データの存在を示すデータである。load_non_intra_quantiser_matrixは、非イントラMB用量子化マトリックス・データの存在を示すデータである。intra_quantiser_matrixは、イントラMB用量子化マトリックスの値を示すデータである。non_intra_quantiser_matrixは、非イントラMB用量子化マトリックスの値を表すデータである。

【0275】

最終段の符号化処理において生成されたビットストリームのピクチャ層のユーザデータエリアには、過去の符号化処理において使用されたシーケンス層のシーケンスエクステンションを表わすデータ要素が、履歴ストリームとして記述される。

【0276】

この過去の符号化処理で使用したシーケンスエクステンション（sequence_extension）を表わすデータ要素は、extension_start_code、extension_start_code_identifier、sequence_extension_present_flag、profile_and_level_indication、progressive_sequence、chroma_format、horizontal_size_extension、vertical_size_extension、bit_rate_extension、vbv_buffer_size_extension、low_delay、frame_rate_extension_n、及びframe_rate_extension_d等のデータ要素である。

【0277】

extension_start_codeは、エクステンションデータのスタート同期コードを表すデータである。extension_start_code_identifierは、どの拡張データが送られるかを示すデータである。sequence_extension_present_flagは、シーケンスエクステンション内のデータが有効であるか無効であるかを示すデータである。profile_and_level_indicationは、ビデオデータのプロファイルとレベルを指定するためのデータである。progressive_sequ

enceは、ビデオデータが順次走査であることを示すデータである。chroma_formatは、ビデオデータの色差フォーマットを指定するためのデータである。

【0278】

horizontal_size_extensionは、シーケンスヘッダのhorizontal_size_valueに加える上位2ビットのデータである。vertical_size_extensionは、シーケンスヘッダのvertical_size_valueに加える上位2ビットのデータである。bit_rate_extensionは、シーケンスヘッダのbit_rate_valueに加える上位12ビットのデータである。vbv_buffer_size_extensionは、シーケンスヘッダのvbv_buffer_size_valueに加える上位8ビットのデータである。low_delayは、Bピクチャを含まないことを示すデータである。frame_rate_extension_nは、シーケンスヘッダのframe_rate_codeと組み合わせてフレームレートを取得するためのデータである。frame_rate_extension_dは、シーケンスヘッダのframe_rate_codeと組み合わせてフレームレートを取得するためのデータである。

10

【0279】

続いて、ビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において使用されたシーケンス層のシーケンスディスプレイエクステンションを表わすデータエレメントが、履歴ストリームとして記述される。

【0280】

このシーケンスディスプレイエクステンション(sequence_display_extension)として記述されているデータエレメントは、extension_start_code、extension_start_code_identifier、sequence_display_extension_present_flag、video_format、colour_description、colour_primaries、transfer_characteristics、matrix_coefficients、display_horizontal_size、及びdisplay_vertical_sizeから構成される。

20

【0281】

extension_start_codeは、エクステンションデータのスタート同期コードを表すデータである。extension_start_code_identifierは、どの拡張データが送られるかを示すコードである。sequence_display_extension_present_flagは、シーケンスディスプレイエクステンション内のデータエレメントが有効か無効かを示すデータである。video_formatは、原信号の映像フォーマットを表すデータである。color_descriptionは、色空間の詳細データがあることを示すデータである。color_primariesは、原信号の色特性の詳細を示すデータである。transfer_characteristicsは、光電変換がどのように行われたのかの詳細を示すデータである。matrix_coefficientsは、原信号が光の三原色からどのように変換されたかの詳細を示すデータである。display_horizontal_sizeは、意図するディスプレイの活性領域(水平サイズ)を表すデータである。display_vertical_sizeは、意図するディスプレイの活性領域(垂直サイズ)を表すデータである。

30

【0282】

続いて、最終段の符号化処理において生成されたビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において生成されたマクロブロックの位相情報(マクロブロックアサインメントデータ(macroblock_assignment_in_user_data))が、履歴ストリームとして記述される。

【0283】

このマクロブロックの位相情報(マクロブロックアサインメントデータ(macroblock_assignment_in_user_data))は、macroblock_assignment_present_flag、v_phase、h_phase等のデータエレメントから構成される。

40

【0284】

このmacroblock_assignment_present_flagは、macroblock_assignment_in_user_data内のデータエレメントが有効か無効かを示すデータである。

【0285】

v_phaseは、画像データからマクロブロックを切り出す際の垂直方向の位相情報(マクロブロックアサインメントデータ(macroblock_assignment_in_user_data))を示すデータである。

【0286】

50

h_phaseは、画像データからマクロブロックを切り出す際の水平方向の位相情報を示すデータである。

【0287】

続いて、最終段の符号化処理によって生成されたビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において使用されたGOP層のGOPヘッダを表わすデータエレメントが、履歴ストリームとして記述されている。

【0288】

このGOPヘッダ (group_of_picture_header) を表わすデータエレメントは、group_start_code、group_of_picture_header_present_flag、time_code、closed_gop、及びbroken_linkから構成される。

10

【0289】

group_start_codeは、GOP層の開始同期コードを示すデータである。group_of_picture_header_present_flagは、group_of_picture_header内のデータエレメントが有効であるか無効であるかを示すデータである。time_codeは、GOPの先頭ピクチャのシーケンスの先頭からの時間を示すタイムコードである。closed_gopは、GOP内の画像が他のGOPから独立再生可能なことを示すフラグデータである。broken_linkは、編集などのためにGOP内の先頭のBピクチャが正確に再生できないことを示すフラグデータである。

【0290】

続いて、最終段の符号化処理によって生成されたビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において使用されたピクチャ層のピクチャヘッダを表わすデータエレメントが、履歴ストリームとして記述されている。

20

【0291】

このピクチャヘッダ (picture_header) に関するデータエレメントは、picture_start_code、temporal_reference、picture_coding_type、vbv_delay、full_pel_forward_vector、forward_f_code、full_pel_backward_vector、及び backward_f_codeから構成される。

【0292】

具体的には、picture_start_codeは、ピクチャ層の開始同期コードを表すデータである。temporal_referenceは、ピクチャの表示順を示す番号でGOPの先頭でリセットされるデータである。picture_coding_typeは、ピクチャタイプを示すデータである。vbv_delayは、ランダムアクセス時の仮想バッファの初期状態を示すデータである。full_pel_forward_vectorは、順方向動きベクトルの精度が整数単位か半画素単位かを示すデータである。forward_f_codeは、順方向動きベクトル探索範囲を表すデータである。full_pel_backward_vectorは、逆方向動きベクトルの精度が整数単位か半画素単位かを示すデータである。backward_f_codeは、逆方向動きベクトル探索範囲を表すデータである。

30

【0293】

続いて、最終段の符号化処理によって生成されたビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において使用されたピクチャ層のピクチャコーディングエクステンションが、履歴ストリームとして記述されている。

【0294】

このピクチャコーディングエクステンション (picture_coding_extension) に関するデータエレメントは、extension_start_code、extension_start_code_identifier、f_code[0][0]、f_code[0][1]、f_code[1][0]、f_code[1][1]、intra_dc_precision、picture_structure、top_field_first、frame_predictive_frame_dct、concealment_motion_vectors、q_scale_type、intra_vlc_format、alternate_scan、repeat_first_field、chroma_420_type、progressive_frame、composite_display_flag、v_axis、field_sequence、sub_carrier、burst_amplitude、及びsub_carrier_phaseから構成される。

40

【0295】

extension_start_codeは、ピクチャ層のエクステンションデータのスタートを示す開始コードである。extension_start_code_identifierは、どの拡張データが送られるかを示

50

すコードである。f_code[0][0]は、フォワード方向の水平動きベクトル探索範囲を表すデータである。f_code[0][1]は、フォワード方向の垂直動きベクトル探索範囲を表すデータである。f_code[1][0]は、バックワード方向の水平動きベクトル探索範囲を表すデータである。f_code[1][1]は、バックワード方向の垂直動きベクトル探索範囲を表すデータである。

【0296】

intra_dc_precisionは、DC係数の精度を表すデータである。picture_structureは、フレームストラクチャかフィールドストラクチャかを示すデータである。フィールドストラクチャの場合は、上位フィールドか下位フィールドかもあわせて示すデータである。top_field_firstは、フレームストラクチャの場合、最初のフィールドが上位か下位かを示すデータである。frame_predictive_frame_dctは、フレーム・ストラクチャの場合、フレーム・モードDCTの予測がフレーム・モードだけであることを示すデータである。concealment_motion_vectorsは、イントラマクロブロックに伝送エラーを隠蔽するための動きベクトルがついていることを示すデータである。

10

【0297】

q_scale_typeは、線形量子化スケールを利用するか、非線形量子化スケールを利用するかを示すデータである。intra_vlc_formatは、イントラマクロブロックに、別の2次元VLCを使うかどうかを示すデータである。alternate_scanは、ジグザグスキャンを使うか、オルタネート・スキャンを使うかの選択を表すデータである。repeat_first_fieldは、2：3プルダウンの際に使われるデータである。chroma_420_typeは、信号フォーマットが4：2：0の場合、次のprogressive_frameと同じ値、そうでない場合は0を表すデータである。progressive_frameは、このピクチャが、順次走査できているかどうかを示すデータである。composite_display_flagは、ソース信号がコンポジット信号であったかどうかを示すデータである。

20

【0298】

v_axisは、ソース信号が、PALの場合に使われるデータである。field_sequenceは、ソース信号が、PALの場合に使われるデータである。sub_carrierは、ソース信号が、PALの場合に使われるデータである。burst_amplitudeは、ソース信号が、PALの場合に使われるデータである。sub_carrier_phaseは、ソース信号が、PALの場合に使われるデータである。

30

【0299】

続いて、最終段の符号化処理によって生成されたビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において使用された量子化マトリックスエクステンションが、履歴ストリームとして記述されている。

【0300】

この量子化マトリックスエクステンション (quant_matrix_extension) に関するデータエレメントは、extension_start_code、extension_start_code_identifier、quant_matrix_extension_present_flag、load_intra_quantiser_matrix、intra_quantiser_matrix[64]、load_non_intra_quantiser_matrix、non_intra_quantiser_matrix[64]、load_chroma_intra_quantiser_matrix、chroma_intra_quantiser_matrix[64]、load_chroma_non_intra_quantiser_matrix、及びchroma_non_intra_quantiser_matrix[64] から構成される。

40

【0301】

extension_start_codeは、この量子化マトリックスエクステンションのスタートを示す開始コードである。extension_start_code_identifierは、どの拡張データが送られるかを示すコードである。quant_matrix_extension_present_flagは、この量子化マトリックスエクステンション内のデータエレメントが有効か無効かを示すためのデータである。load_intra_quantiser_matrixは、イントラマクロブロック用の量子化マトリックスデータの存在を示すデータである。intra_quantiser_matrixは、イントラマクロブロック用の量子化マトリックスの値を示すデータである。

【0302】

50

load_non_intra_quantiser_matrixは、非イントラマクロブロック用の量子化マトリックスデータの存在を示すデータである。non_intra_quantiser_matrixは、非イントラマクロブロック用の量子化マトリックスの値を表すデータである。load_chroma_intra_quantiser_matrixは、色差イントラマクロブロック用の量子化マトリックス・データの存在を示すデータである。chroma_intra_quantiser_matrixは、色差イントラマクロブロック用の量子化マトリックスの値を示すデータである。load_chroma_non_intra_quantiser_matrixは、色差非イントラマクロブロック用の量子化マトリックス・データの存在を示すデータである。chroma_non_intra_quantiser_matrixは、色差非イントラマクロブロック用の量子化マトリックスの値を示すデータである。

【0303】

10

続いて、最終段の符号化処理によって生成されたビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において使用されたコピーライトエクステンションが、履歴ストリームとして記述されている。

【0304】

このコピーライトエクステンション (copyright_extension) に関するデータエレメントは、extension_start_code、extension_start_code_identifier、copyright_extension_present_flag、copyright_flag、copyright_identifier、original_or_copy、copyright_number_1、copyright_number_2、及び copyright_number_3から構成される。

【0305】

extension_start_codeは、コピーライトエクステンションのスタートを示す開始コードである。extension_start_code_identifierのどのエクステンションデータが送られるかを示すコードである。copyright_extension_present_flagは、このコピーライトエクステンション内のデータエレメントが有効か無効かを示すためのデータである。copyright_flagは、次のコピーライトエクステンション又はシーケンスエンドまで、符号化されたビデオデータに対してコピー権が与えられているか否かを示す。

20

【0306】

copyright_identifierは、ISO/IEC JTC/SC29によって指定されたコピー権の登録機関を識別するためのデータである。original_or_copyは、ビットストリーム中のデータが、オリジナルデータであるかコピーデータであるかを示すデータである。copyright_number_1は、コピーライトナンバーのビット44から63を表わすデータである。copyright_number_2は、コピーライトナンバーのビット22から43を表わすデータである。copyright_number_3は、コピーライトナンバーのビット0から21を表わすデータである。

30

【0307】

続いて、最終段の符号化処理によって生成されたビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において使用されたピクチャディスプレイエクステンション (picture_display_extension) が、履歴ストリームとして記述されている。

【0308】

このピクチャディスプレイエクステンションを表わすデータエレメントは、extension_start_code、extension_start_code_identifier、picture_display_extension_present_flag、frame_center_horizontal_offset_1、frame_center_vertical_offset_1、frame_center_horizontal_offset_2、frame_center_vertical_offset_2、frame_center_horizontal_offset_3、及びframe_center_vertical_offset_3から構成される。

40

【0309】

extension_start_codeは、ピクチャディスプレイエクステンションのスタートを示すための開始コードである。extension_start_code_identifierは、どの拡張データが送られるかを示すコードである。picture_display_extension_present_flagは、ピクチャディスプレイエクステンション内のデータエレメントが有効か無効かを示すデータである。frame_center_horizontal_offsetは、表示エリアの水平方向のオフセットを示すデータであって、3つのオフセット値まで定義することができる。frame_center_vertical_offsetは、表示エリアを垂直方向のオフセットを示すデータであって、3つのオフセット値まで定義

50

することができる。

【0310】

本発明の特徴的な点として、このピクチャディスプレイエクステンションを表わす履歴情報の次に、re_coding_stream_information に関するデータエレメントが記述されている。re_coding_stream_information はre_coding_stream_information に関するデータエレメントは、user_data_start_code, re_coding_stream_info_ID, red_bw_flag, red_bw_indicatorなどのデータエレメントより構成される。

【0311】

user_data_start_codeは、user_dataが開始することを表すスタートコードである。re_coding_stream_info_IDは、16ビットの整数であり、re_coding_stream_info()関数の識別のために用いられる。その値は、具体的には、"1001 0001 1110 1100" (0x91ec) とされる。

10

【0312】

red_bw_flagは、1ビットのフラグであり、全ての履歴情報に関するコーディングパラメータを伝送する場合には0とされ、履歴情報に関するコーディングパラメータが選択的に伝送する場合には1とされる。red_bw_indicatorは、2ビットの整数であり、コーディングパラメータのデータセットを定義するためのインジケータである。

【0313】

このre_coding_stream_information、red_bw_flag、red_bw_indicator及びデータセットに関しては、詳しくは後述する。

20

【0314】

最終段の符号化処理において生成されたビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において使用されたユーザデータ (user_data) が、履歴ストリームとして記述されている。

【0315】

このユーザデータの次には、過去の符号化処理において使用されたマクロブロック層に関する情報が、履歴ストリームとして記述されている。

【0316】

このマクロブロック層に関する情報は、macroblock_address_h、macroblock_address_v、slice_header_present_flag、skipped_macroblock_flag等のマクロブロック (macroblock) の位置に関するデータエレメントと、macroblock_quant、macroblock_motion_forward、macroblock_motion_backward、macroblock_pattern、macroblock_intra、spatial_temporal_weight_code_flag、frame_motion_type、及びdct_type等のマクロブロックモード (macroblock_modes[]) に関するデータエレメントと、quantiser_scale_code等の量子化ステップ制御に関するデータエレメントと、PMV[0][0][0]、PMV[0][0][1]、motion_vertical_field_select[0][0]、PMV[0][1][0]、PMV[0][1][1]、motion_vertical_field_select[0][1]、PMV[1][0][0]、PMV[1][0][1]、motion_vertical_field_select[1][0]、PMV[1][1][0]、PMV[1][1][1]、motion_vertical_field_select[1][1]等の動き補償に関するデータエレメントと、coded_block_pattern等のマクロブロックパターンに関するデータエレメントと、num_mv_bits、num_coef_bits、及びnum_other_bits等の発生符号量に関するデータエレメントから構成されている。

30

40

【0317】

以下にマクロブロック層に関するデータエレメントについて詳細に説明する。

【0318】

macroblock_address_hは、現在のマクロブロックの水平方向の絶対位置を定義するためのデータである。macroblock_address_vは、現在のマクロブロックの垂直方向の絶対位置を定義するためのデータである。slice_header_present_flagは、このマクロブロックがスライス層の先頭であり、スライスヘッダを伴うか否かを示すデータである。skipped_macroblock_flagは、復号処理においてこのマクロブロックをスキップするか否かを示すデータである。

50

【0319】

macroblock_quantは、後述する図63と図64に示されたマクロブロックタイプ (macroblock_type) から導かれるデータであって、quantiser_scale_codeがビットストリーム中に現れるか否かを示すデータである。macroblock_motion_forwardは、図63と図64に示されたマクロブロックタイプから導かれるデータであって、復号処理で使用されるデータである。macroblock_motion_backwardは、図63と図64に示されたマクロブロックタイプから導かれるデータであって、復号処理で使用されるデータである。macroblock_patternは、図63と図64に示されたマクロブロックタイプから導かれるデータであって、coded_block_patternがビットストリーム中に現れるか否かを示すデータである。

【0320】

macroblock_intraは、図63と図64に示されたマクロブロックタイプから導かれるデータであって、復号処理で使用されるデータである。spatial_temporal_weight_code_flagは、図63と図64に示されたマクロブロックタイプから導かれるデータであって、時間スケラビリティで下位レイヤ画像のアップサンプリング方法を示すspatial_temporal_weight_codeは、ビットストリーム中に存在するか否かを示すデータである。

【0321】

frame_motion_typeは、フレームのマクロブロックの予測タイプを示す2ビットのコードである。予測ベクトルが2個でフィールドベースの予測タイプであれば「00」であって、予測ベクトルが1個でフィールドベースの予測タイプであれば「01」であって、予測ベクトルが1個でフレームベースの予測タイプであれば「10」であって、予測ベクトルが1個でディアルプライムの予測タイプであれば「11」である。field_motion_typeは、フィールドのマクロブロックの動き予測を示す2ビットのコードである。予測ベクトルが1個でフィールドベースの予測タイプであれば「01」であって、予測ベクトルが2個で18×8マクロブロックベースの予測タイプであれば「10」であって、予測ベクトルが1個でディアルプライムの予測タイプであれば「11」である。dct_typeは、DCTがフレームDCTモードか、フィールドDCTモードかを示すデータである。quantiser_scale_codeはマクロブロックの量子化ステップサイズを示すデータである。

【0322】

次に動きベクトルに関するデータエレメントについて説明する。動きベクトルは、復号時に必要な動きベクトルを減少させるために、先に符号化されたベクトルに関し差分として符号化される。動きベクトルの復号を行うために復号器は、4個の動きベクトル予測値（それぞれ水平及び垂直成分を伴う）を維持しなければならない。この予測動きベクトルをPMV[r][s][v]と表わすことにしている。[r]は、マクロブロックにおける動きベクトルが第1のベクトルであるのか、第2のベクトルであるのかを示すフラグであって、マクロブロックにおけるベクトルが第1のベクトルである場合には「0」となって、マクロブロックにおけるベクトルが第2のベクトルである場合には「1」となる。[s]は、マクロブロックにおける動きベクトルの方向が、前方向であるのか後方向であるのかを示すフラグであって、前方向動きベクトルの場合には「0」となって、後方向動きベクトルの場合には「1」となる。[v]は、マクロブロックにおけるベクトルの成分が、水平方向であるのか垂直方向であるのかを示すフラグであって、水平方向成分の場合には「0」となって、垂直方向成分の場合には「1」となる。

【0323】

従って、PMV[0][0][0]は、第1のベクトルの前方向の動きベクトルの水平方向成分のデータを表わし、PMV[0][0][1]は、第1のベクトルの前方向の動きベクトルの垂直方向成分のデータを表わし、PMV[0][1][0]は、第1のベクトルの後方向の動きベクトルの水平方向成分のデータを表わし、PMV[0][1][1]は、第1のベクトルの後方向の動きベクトルの垂直方向成分のデータを表わし、PMV[1][0][0]は、第2のベクトルの前方向の動きベクトルの水平方向成分のデータを表わし、PMV[1][0][1]は、第2のベクトルの前方向の動きベクトルの垂直方向成分のデータを表わし、PMV[1][1][0]は、第2のベクトルの後方向の動きベクトルの水平方向成分のデータを表わし、PMV[1][1][1]は、第2のベクトルの後方向の動

きベクトルの垂直方向成分のデータを表わしている。

【0324】

motion_vertical_field_select[r][s]は、予測の形式にいずれの参照フィールドを使用するかを示すデータである。このmotion_vertical_field_select[r][s]が「0」の場合には、トップ参照フィールドを使用し、「1」の場合には、ボトム参照フィールドを使用することを示している。

【0325】

よって、motion_vertical_field_select[0][0]は、第1のベクトルの前方向の動きベクトルを生成する際の参照フィールドを示し、motion_vertical_field_select[0][1]は、第1のベクトルの後方向の動きベクトルを生成する際の参照フィールドを示し、motion_ver
tical_field_select[1][0]は、第2のベクトルの前方向の動きベクトルを生成する際の参
照フィールドを示し、motion_vertical_field_select[1][1]は、第2ベクトルの後方向の
動きベクトルを生成する際の参照フィールドを示している。

10

【0326】

coded_block_patternは、DCT係数を格納する複数のDCTブロックのうち、どのDCTブロッ
クに、有意係数（非0係数）があるかを示す可変長のデータである。num_mv_bitsは、マ
クロブロック中の動きベクトルの符号量を示すデータである。num_coef_bitsは、マクロ
ブロック中のDCT係数の符号量を示すデータである。num_other_bitsは、マクロブロック
の符号量で、動きベクトル及びDCT係数以外の符号量を示すデータである。

【0327】

次に、可変長の履歴ストリームから各データエレメントをデコードするためのシNTAX
クスについて、図47乃至図67を参照して説明する。

20

【0328】

この可変長の履歴ストリームは、next_start_code()関数、sequence_header()関数、se
quence_extension()関数、extension_and_user_data(0)関数、group_of_picture_header(
)関数、extension_and_user_data(1)関数、picture_header()関数、picture_coding_exte
nsion()関数、re_coding_stream_info()関数、extension_and_user_data(2)関数、及びpi
cture_data()関数によって定義されたデータエレメントによって構成される。

【0329】

next_start_code()関数は、ビットストリーム中に存在するスタートコードを探すため
の関数であるので、履歴ストリームの最も先頭には、図48に示すような、過去の符号化
処理において使用されたデータエレメントであってsequence_header()関数によって定義
されたデータエレメントが記述されている。

30

【0330】

sequence_header()関数によって定義されたデータエレメントは、sequence_header_cod
e、sequence_header_present_flag、horizontal_size_value、vertical_size_value、asp
ect_ratio_information、frame_rate_code、bit_rate_value、marker_bit、VBV_buffer_s
ize_value、constrained_parameter_flag、load_intra_quantiser_matrix、intra_quant
iser_matrix、load_non_intra_quantiser_matrix、及びnon_intra_quantiser_matrix等で
ある。

40

【0331】

sequence_header_codeは、シーケンス層のスタート同期コードを表すデータである。se
quence_header_present_flagは、sequence_header内のデータが有効か無効かを示すデー
タである。horizontal_size_valueは、画像の水平方向の画素数の下位12ビットから成る
データである。vertical_size_valueは、画像の縦のライン数の下位12ビットからなるデ
ータである。aspect_ratio_informationは、画素のアスペクト比（縦横比）または表示画
面アスペクト比を表すデータである。frame_rate_codeは、画像の表示周期を表すデー
タである。bit_rate_valueは、発生ビット量に対する制限のためのビット・レートの下位18
ビット(400bsp単位で切り上げる)データである。

【0332】

50

marker_bitは、スタートコードエミュレーションを防止するために挿入されるビットデータである。VBV_buffer_size_valueは、発生符号量制御用の仮想バッファ（ビデオバッファペリファイヤー）の大きさを決める値の下位10ビットデータである。constrained_parameter_flagは、各パラメータが制限以内であることを示すデータである。load_intra_quantiser_matrixは、イントラMB用量子化マトリックス・データの存在を示すデータである。intra_quantiser_matrixは、イントラMB用量子化マトリックスの値を示すデータである。load_non_intra_quantiser_matrixは、非イントラMB用量子化マトリックス・データの存在を示すデータである。non_intra_quantiser_matrixは、非イントラMB用量子化マトリックスの値を表すデータである。

【0333】

sequence_header()関数によって定義されたデータエレメントの次には、図49で示すような、sequence_extension()関数によって定義されたデータエレメントが、履歴ストリームとして記述されている。

【0334】

sequence_extension()関数によって定義されたデータエレメントとは、extension_start_code、extension_start_code_identifier、sequence_extension_present_flag、profile_and_level_indication、progressive_sequence、chroma_format、horizontal_size_extension、vertical_size_extension、bit_rate_extension、vbv_buffer_size_extension、low_delay、frame_rate_extension_n、及びframe_rate_extension_d等のデータエレメントである。

【0335】

extension_start_codeは、エクステンションデータのスタート同期コードを表すデータである。extension_start_code_identifierは、どの拡張データが送られるかを示すデータである。sequence_extension_present_flagは、シーケンスエクステンション内のデータが有効であるか無効であることを示すデータである。profile_and_level_indicationは、ビデオデータのプロファイルとレベルを指定するためのデータである。progressive_sequenceは、ビデオデータが順次走査であることを示すデータである。chroma_formatは、ビデオデータの色差フォーマットを指定するためのデータである。horizontal_size_extensionは、シーケンスヘッダのhorizontal_size_valueに加える上位2ビットのデータである。vertical_size_extensionは、シーケンスヘッダのvertical_size_valueに加える上位2ビットのデータである。bit_rate_extensionは、シーケンスヘッダのbit_rate_valueに加える上位12ビットのデータである。vbv_buffer_size_extensionは、シーケンスヘッダのvbv_buffer_size_valueに加える上位8ビットのデータである。

【0336】

low_delayは、Bピクチャを含まないことを示すデータである。frame_rate_extension_nは、シーケンスヘッダのframe_rate_codeと組み合わせてフレームレートを得るためのデータである。frame_rate_extension_dは、シーケンスヘッダのframe_rate_codeと組み合わせてフレームレートを得るためのデータである。

【0337】

sequence_extension()関数によって定義されたデータエレメントの次には、図50に示すようなextension_and_user_data(0)関数によって定義されたデータエレメントが、履歴ストリームとして記述されている。extension_and_user_data(i)関数は、「i」が1以外のときは、extension_data()関数によって定義されるデータエレメントは記述せずに、user_data()関数によって定義されるデータエレメントのみを履歴ストリームとして記述する。よって、extension_and_user_data(0)関数は、user_data()関数によって定義されるデータエレメントのみを履歴ストリームとして記述する。

【0338】

user_data()関数は、図51に示されたようなシンタックスに基いて、ユーザデータを履歴ストリームとして記述する。

【0339】

extension_and_user_data(0)関数によって定義されたデータエレメントの次には、図 5 2 に示すようなgroup_of_picture_header()関数によって定義されたデータエレメント、及びextension_and_user_data(1)関数によって定義されるデータエレメントが、履歴ストリームとして記述されている。但し、履歴ストリーム中に、GOP層のスタートコードを示すgroup_start_codeが記述されている場合にのみ、group_of_picture_header()関数によって定義されたデータエレメント、及びextension_and_user_data(1)関数によって定義されるデータエレメントが記述されている。

【0 3 4 0】

group_of_picture_header()関数によって定義されたデータエレメントは、group_start_code、group_of_picture_header_present_flag、time_code、closed_gop、及びbroken_l 10
inkから構成される。

【0 3 4 1】

group_start_codeは、GOP層の開始同期コードを示すデータである。group_of_picture_header_present_flagは、group_of_picture_header内のデータエレメントが有効であるか無効であることを示すデータである。time_codeは、GOPの先頭ピクチャのシーケンスの先頭からの時間を示すタイムコードである。closed_gopは、GOP内の画像が他のGOPから独立再生可能なことを示すフラグデータである。broken_linkは、編集などのためにGOP内の先頭のBピクチャが正確に再生できないことを示すフラグデータである。

【0 3 4 2】

extension_and_user_data(1)関数は、extension_and_user_data(0)関数と同じように、user_data()関数によって定義されるデータエレメントのみを履歴ストリームとして記述する。 20

【0 3 4 3】

もし、履歴ストリーム中に、GOP層のスタートコードを示すgroup_start_codeが存在しない場合には、これらのgroup_of_picture_header()関数及びextension_and_user_data(1)関数によって定義されるデータエレメントは、履歴ストリーム中には記述されていない。その場合には、extension_and_user_data(0)関数によって定義されたデータエレメントの次に、picture_headr()関数によって定義されたデータエレメントが履歴ストリームとして記述されている。

【0 3 4 4】

picture_headr()関数によって定義されたデータエレメントは、図 5 3 に示すように、picture_start_code、temporal_reference、picture_coding_type、vbm_delay、full_pel_forward_vector、forward_f_code、full_pel_backward_vector、backward_f_code、extra_bit_picture、及びextra_information_pictureである。 30

【0 3 4 5】

具体的には、picture_start_codeは、ピクチャ層の開始同期コードを表すデータである。temporal_referenceは、ピクチャの表示順を示す番号でGOPの先頭でリセットされるデータである。picture_coding_typeは、ピクチャタイプを示すデータである。vbm_delayは、ランダムアクセス時の仮想バッファの初期状態を示すデータである。full_pel_forward_vectorは、順方向動きベクトルの精度が整数単位か半画素単位かを示すデータである。forward_f_codeは、順方向動きベクトル探索範囲を表すデータである。full_pel_backward_vectorは、逆方向動きベクトルの精度が整数単位か半画素単位かを示すデータである。backward_f_codeは、逆方向動きベクトル探索範囲を表すデータである。extra_bit_pictureは、後続する追加情報の存在を示すフラグである。このextra_bit_pictureが「1」の場合には、次にextra_information_pictureが存在し、extra_bit_pictureが「0」の場合には、これに続くデータが無いことを示している。extra_information_pictureは、規格において予約された情報である。 40

【0 3 4 6】

picture_headr()関数によって定義されたデータエレメントの次には、図 5 4 に示すようなpicture_coding_extension()関数によって定義されたデータエレメントが、履歴スト 50

リームとして記述されている。

【0347】

このpicture_coding_extension()関数によって定義されたデータエレメントとは、extension_start_code、extension_start_code_identifier、f_code[0][0]、f_code[0][1]、f_code[1][0]、f_code[1][1]、intra_dc_precision、picture_structure、top_field_first、frame_predictive_frame_dct、concealment_motion_vectors、q_scale_type、intra_vlc_format、alternate_scan、repeat_firt_field、chroma_420_type、progressive_frame、composite_display_flag、v_axis、field_sequence、sub_carrier、burst_amplitude、及びsub_carrier_phaseから構成される。

【0348】

extension_start_codeは、ピクチャ層のエクステンションデータのスタートを示す開始コードである。extension_start_code_identifierは、どの拡張データが送られるかを示すコードである。f_code[0][0]は、フォワード方向の水平動きベクトル探索範囲を表すデータである。f_code[0][1]は、フォワード方向の垂直動きベクトル探索範囲を表すデータである。f_code[1][0]は、バックワード方向の水平動きベクトル探索範囲を表すデータである。f_code[1][1]は、バックワード方向の垂直動きベクトル探索範囲を表すデータである。intra_dc_precisionは、DC係数の精度を表すデータである。

【0349】

picture_structureは、フレームストラクチャかフィールドストラクチャかを示すデータである。フィールドストラクチャの場合は、上位フィールドか下位フィールドかもあわせて示すデータである。top_field_firstは、フレームストラクチャの場合、最初のフィールドが上位か下位かを示すデータである。frame_predictive_frame_dctは、フレーム・ストラクチャの場合、フレーム・モードDCTの予測がフレーム・モードだけであることを示すデータである。concealment_motion_vectorsは、イントラマクロブロックに伝送エラーを隠蔽するための動きベクトルがついていることを示すデータである。q_scale_typeは、線形量子化スケールを利用するか、非線形量子化スケールを利用するかを示すデータである。intra_vlc_formatは、イントラマクロブロックに、別の2次元VLCを使うかどうかを示すデータである。

【0350】

alternate_scanは、ジグザグスキャンを使うか、オルタネート・スキャンを使うかの選択を表すデータである。repeat_firt_fieldは、2：3プルダウンの際に使われるデータである。chroma_420_typeは、信号フォーマットが4：2：0の場合、次のprogressive_frameと同じ値、そうでない場合は0を表すデータである。progressive_frameは、このピクチャが、順次走査できているかどうかを示すデータである。composite_display_flagは、ソース信号がコンポジット信号であったかどうかを示すデータである。v_axisは、ソース信号が、PALの場合に使われるデータである。field_sequenceは、ソース信号が、PALの場合に使われるデータである。sub_carrierは、ソース信号が、PALの場合に使われるデータである。burst_amplitudeは、ソース信号が、PALの場合に使われるデータである。sub_carrier_phaseは、ソース信号が、PALの場合に使われるデータである。

【0351】

picture_coding_extension()関数によって定義されたデータエレメントの次には、re_coding_stream_info()関数によって定義されたデータエレメントが履歴ストリームとして記述されている。このre_coding_stream_info()関数は、本発明の特徴的な関数であって、主に履歴情報の組み合わせを記述する場合に用いられるものであり、その詳細については、図68を参照して後述する。

【0352】

re_coding_stream_info()関数によって定義されたデータエレメントの次には、extensions_and_user_data(2)によって定義されたデータエレメントが、履歴ストリームとして記述されている。このextension_and_user_data(2)関数は、図50に示したように、ビットストリーム中にエクステンションスタートコード(extension_start_code)が存在する場

合には、extension_data()関数によって定義されるデータエレメントが記述されている。このデータエレメントの次には、ビットストリーム中にユーザデータスタートコード (user_data_start_code) が存在する場合には、user_data()関数によって定義されるデータエレメントが記述されている。但し、ビットストリーム中にエクステンションスタートコード及びユーザデータスタートコードが存在しない場合には extension_data()関数 及び user_data()関数によって定義されるデータエレメントはビットストリーム中には記述されていない。

【0353】

extension_data()関数は、図55に示すように、extension_start_codeを示すデータエレメントと、quant_matrix_extension()関数、copyright_extension()関数、及びpicture_display_extension()関数によって定義されるデータエレメントとを、ビットストリーム中に履歴ストリームとして記述するための関数である。

10

【0354】

quant_matrix_extension()関数によって定義されるデータエレメントは、図56に示すように、extension_start_code、extension_start_code_identifier、quant_matrix_extension_present_flag、load_intra_quantiser_matrix、intra_quantiser_matrix[64]、load_non_intra_quantiser_matrix、non_intra_quantiser_matrix[64]、load_chroma_intra_quantiser_matrix、chroma_intra_quantiser_matrix[64]、load_chroma_non_intra_quantiser_matrix、及びchroma_non_intra_quantiser_matrix[64]である。

20

【0355】

extension_start_codeは、この量子化マトリックスエクステンションのスタートを示す開始コードである。extension_start_code_identifierは、どの拡張データが送られるかを示すコードである。quant_matrix_extension_present_flagは、この量子化マトリックスエクステンション内のデータエレメントが有効か無効かを示すためのデータである。load_intra_quantiser_matrixは、イントラマクロブロック用の量子化マトリックスデータの存在を示すデータである。intra_quantiser_matrixは、イントラマクロブロック用の量子化マトリックスの値を示すデータである。

【0356】

load_non_intra_quantiser_matrixは、非イントラマクロブロック用の量子化マトリックスデータの存在を示すデータである。non_intra_quantiser_matrixは、非イントラマクロブロック用の量子化マトリックスの値を表すデータである。load_chroma_intra_quantiser_matrixは、色差イントラマクロブロック用の量子化マトリックス・データの存在を示すデータである。chroma_intra_quantiser_matrixは、色差イントラマクロブロック用の量子化マトリックスの値を示すデータである。load_chroma_non_intra_quantiser_matrixは、色差非イントラマクロブロック用の量子化マトリックス・データの存在を示すデータである。chroma_non_intra_quantiser_matrixは、色差非イントラマクロブロック用の量子化マトリックスの値を示すデータである。

30

【0357】

copyright_extension()関数によって定義されるデータエレメントは、図57に示すように、extension_start_code、extension_start_code_identifier、copyright_extension_present_flag、copyright_flag、copyright_identifier、original_or_copy、copyright_number_1、copyright_number_2、及び copyright_number_3から構成される。

40

【0358】

extension_start_codeは、コピーライトエクステンションのスタートを示す開始コードである。extension_start_code_identifierどのエクステンションデータが送られるかを示すコードである。copyright_extension_present_flagは、このコピーライトエクステンション内のデータエレメントが有効か無効かを示すためのデータである。

【0359】

copyright_flagは、次のコピーライトエクステンション又はシーケンスエンドまで、符号化されたビデオデータに対してコピー権が与えられているか否かを示す。copyright_id

50

entifierは、ISO/IEC JTC/SC29によって指定されたコピー権の登録機関を識別するためのデータである。original_or_copyは、ビットストリーム中のデータが、オリジナルデータであるかコピーデータであるかを示すデータである。copyright_number_1は、コピーライトナンバーのビット44から63を表わすデータである。copyright_number_2は、コピーライトナンバーのビット22から43を表わすデータである。copyright_number_3は、コピーライトナンバーのビット0から21を表わすデータである。

【0360】

picture_display_extension()関数によって定義されるデータエレメントは、図58に示すように、extension_start_code_Identifier、frame_center_horizontal_offset、frame_center_vertical_offset等である。

10

【0361】

extension_start_code_Identifierは、どの拡張データが送られるかを示すコードである。frame_center_horizontal_offsetは、表示エリアの水平方向のオフセットを示すデータであって、number_of_frame_center_offsetsによって定義される数のオフセット値を定義することができる。frame_center_vertical_offsetは、表示エリアを垂直方向のオフセットを示すデータであって、number_of_frame_center_offsetsによって定義される数のオフセット値を定義することができる。

【0362】

再び図47に戻って、extension_and_user_data(2)関数によって定義されるデータエレメントの次には、picture_data()関数によって定義されるデータエレメントが、履歴ストリームとして記述されている。但し、このpicture_data()関数は、red_bw_flagが1ではないか、または、red_bw_indicatorが2以下である場合に存在する。このred_bw_flagとred_bw_indicatorは、re_coding_stream_info()関数に記述されており、これらについては、図68と図69を参照して後述する。

20

【0363】

picture_data()関数によって定義されるデータエレメントは、図59に示すように、slice()関数によって定義されるデータエレメントである。このslice()関数によって定義されるデータエレメントはビットストリーム中に少なくとも1個記述されている。

【0364】

slice()関数は、図60に示されるように、slice_start_code、slice_quantiser_scale_code、intra_slice_flag、intra_slice、reserved_bits、extra_bit_slice、extra_information_slice、及びextra_bit_slice等のデータエレメントと、macroblock()関数によって定義されるデータエレメントを、履歴ストリームとして記述するための関数である。

30

【0365】

slice_start_codeは、slice()関数によって定義されるデータエレメントのスタートを示すスタートコードである。slice_quantiser_scale_codeは、このスライス層に存在するマクロブロックに対して設定された量子化ステップサイズを示すデータである。しかし、各マクロブロック毎に、quantiser_scale_codeが設定されている場合には、各マクロブロックに対して設定されたmacroblock_quantiser_scale_codeのデータが優先して使用される。

40

【0366】

intra_slice_flagは、ビットストリーム中にintra_slice及びreserved_bitsが存在するか否かを示すフラグである。intra_sliceは、スライス層中にノンイントラマクロブロックが存在するか否かを示すデータである。スライス層におけるマクロブロックのいずれかがノンイントラマクロブロックである場合には、intra_sliceは「0」となり、スライス層におけるマクロブロックの全てがノンイントラマクロブロックである場合には、intra_sliceは「1」となる。reserved_bitsは、7ビットのデータであって「0」の値を取る。extra_bit_sliceは、履歴ストリームとして追加の情報が存在することを示すフラグであって、次にextra_information_sliceが存在する場合には「1」に設定される。追加の情報が存在しない場合には「0」に設定される。これらのデータエレメントの次には、macr

50

oblock()関数によって定義されたデータエレメントが、履歴ストリームとして記述されている。

【0367】

macroblock()関数は、図61に示すように、macroblock_escape、macroblock_address_increment、及びmacroblock_quantiser_scale_code、及びmarker_bit等のデータエレメントと、macroblock_modes()関数、motion_vectors(s)関数、及びcode_block_pattern()関数によって定義されたデータエレメントを記述するための関数である。

【0368】

macroblock_escapeは、参照マクロブロックと前のマクロブロックとの水平方向の差が34以上であるか否かを示す固定ビット列である。参照マクロブロックと前のマクロブロックとの水平方向の差が34以上の場合には、macroblock_address_incrementの値に33をプラスする。macroblock_address_incrementは、参照マクロブロックと前のマクロブロックとの水平方向の差を示すデータである。もし、このmacroblock_address_incrementの前にmacroblock_escapeが1つ存在するのであれば、このmacroblock_address_incrementの値に33をプラスした値が、実際の参照マクロブロックと前のマクロブロックとの水平方向の差分を示すデータとなる。

【0369】

macroblock_quantiser_scale_codeは、各マクロブロック毎に設定された量子化ステップサイズであり、macroblock_quantが「1」のときだけ存在する。各スライス層には、スライス層の量子化ステップサイズを示すslice_quantiser_scale_codeが設定されているが、参照マクロブロックに対してmacroblock_quantiser_scale_codeが設定されている場合には、この量子化ステップサイズを選択する。

【0370】

macroblock_address_incrementの次には、macroblock_modes()関数によって定義されるデータエレメントが記述されている。macroblock_modes()関数は、図62に示すように、macroblock_type、frame_motion_type、field_motion_type、dct_type等のデータエレメントを、履歴ストリームとして記述するための関数である。

【0371】

macroblock_typeは、マクロブロックの符号化タイプを示すデータである。その詳細は、図65乃至図67を参照して後述する。

【0372】

もし、macroblock_motion_forward又はmacroblock_motion_backwardが「1」であり、ピクチャ構造がフレームであり、さらにframe_pred_frame_dctが「0」である場合には、macroblock_typeを表わすデータエレメントの次にframe_motion_typeを表わすデータエレメントが記述されている。尚、このframe_pred_frame_dctは、frame_motion_typeがビットストリーム中に存在するか否かを示すフラグである。

【0373】

frame_motion_typeは、フレームのマクロブロックの予測タイプを示す2ビットのコードである。予測ベクトルが2個でフィールドベースの予測タイプであれば「00」であって、予測ベクトルが1個でフィールドベースの予測タイプであれば「01」であって、予測ベクトルが1個でフレームベースの予測タイプであれば「10」であって、予測ベクトルが1個でディアルプライムの予測タイプであれば「11」である。

【0374】

frame_motion_typeを記述する条件が満足されない場合には、macroblock_typeを表わすデータエレメントの次にfield_motion_typeを表わすデータエレメントが記述されている。

【0375】

field_motion_typeは、フィールドのマクロブロックの動き予測を示す2ビットのコードである。予測ベクトルが1個でフィールドベースの予測タイプであれば「01」であって、予測ベクトルが2個で18×8マクロブロックベースの予測タイプであれば「10」

10

20

30

40

50

であって、予測ベクトルが1個でディアルプライムの予測タイプであれば「1 1」である。

【0376】

もし、ピクチャ構造がフレームで、frame_pred_frame_dctがframe_motion_typeがビットストリーム中に存在することを示し、且つ、frame_pred_frame_dctがdct_typeがビットストリーム中に存在することを示している場合には、macroblock_typeを表わすデータエレメントの次にdct_typeを表わすデータエレメントが記述されている。尚、dct_typeは、DCTがフレームDCTモードか、フィールドDCTモードかを示すデータである。

【0377】

再び図61に戻って、もし、参照マクロブロックが前方予測マクロブロックであるか、又は参照マクロブロックがイントラマクロブロックであって且つコンシール処理のマクロブロックのいずれかの場合には、motion_vectors(0)関数によって定義されるデータエレメントが記述される。また、参照マクロブロックが後方予測マクロブロックである場合には、motion_vectors(1)関数によって定義されるデータエレメントが記述される。尚、motion_vectors(0)関数は、第1番目の動きベクトルに関するデータエレメントを記述するための関数であって、motion_vectors(1)関数は、第2番目の動きベクトルに関するデータエレメントを記述するための関数である。

【0378】

motion_vectors(s)関数は、図63に示されるように、動きベクトルに関するデータエレメントを記述するための関数である。

【0379】

もし、動きベクトルが1個でディアルプライム予測モードを使用していない場合には、motion_vertical_field_select[0][s]とmotion_vector(0,s)によって定義されるデータエレメントが記述される。

【0380】

このmotion_vertical_field_select[r][s]は、第1番目の動きベクトル（前方又は後方のどちらのベクトルであっても良い）が、ボトムフィールドを参照して作られたベクトルであるかトップフィールドを参照して作られたベクトルであるかを示すフラグである。この指標“r”は、第1番目のベクトル又は第2番目のベクトルのいずれのベクトルであるかを示す指標であって、“s”は、予測方向が前方又は後方予測のいずれであるかを示す指標である。

【0381】

motion_vector(r,s)関数は、図64に示されるように、motion_code[r][s][t]に関するデータ列と、motion_residual[r][s][t]に関するデータ列と、dmvector[t]を表わすデータとを記述するための関数である。

【0382】

motion_code[r][s][t]は、動きベクトルの大きさを $-16 \sim +16$ の範囲で表わす可変長のデータである。motion_residual[r][s][t]は、動きベクトルの残差を表わす可変長のデータである。よって、このmotion_code[r][s][t]とmotion_residual[r][s][t]との値によって詳細な動きベクトルを記述することができる。dmvector[t]は、ディアルプライム予測モードのときに、一方のフィールド（例えばボトムフィールドに対してトップフィールドを一方のフィールドとする）における動きベクトルを生成するために、時間距離に応じて既存の動きベクトルがスケールされると共に、トップフィールドとボトムフィールドとのライン間の垂直方向のずれを反映させるために垂直方向に対して補正を行うデータである。この指標“r”は、第1番目のベクトル又は第2番目のベクトルのいずれのベクトルであるかを示す指標であって、“s”は、予測方向が前方又は後方予測のいずれであるかを示す指標である。“s”は、動きベクトルが垂直方向の成分であるか水平方向の成分であるかを示すデータである。

【0383】

図64に示されmotion_vector(r,s)関数によって、まず、水平方向のmotion_coder[r][

s][0]を表わすデータ列が、履歴ストリームとして記述される。motion_residual[0][s][t]及びmotion_residual[1][s][t]の双方のビット数は、f_code[s][t]で示されるので、f_code[s][t]が1でない場合には、motion_residual[r][s][t]がビットストリーム中に存在することを示すことになる。水平方向成分のmotion_residual[r][s][0]が「1」でなくて、水平方向成分のmotion_code[r][s][0]が「0」でないということは、ビットストリーム中にmotion_residual[r][s][0]を表わすデータエレメントが存在し、動きベクトルの水平方向成分が存在するということを意味しているので、その場合には、水平方向成分のmotion_residual[r][s][0]を表わすデータエレメントが記述されている。

【0384】

続いて、垂直方向のmotion_coder[r][s][1]を表わすデータ列が、履歴ストリームとして記述される。同じようにmotion_residual[0][s][t]及びmotion_residual[1][s][t]の双方のビット数は、f_code[s][t]で示されるので、f_code[s][t]が1でない場合には、motion_residual[r][s][t]がビットストリーム中に存在することを表わすことになる。motion_residual[r][s][1]が「1」でなくて、motion_code[r][s][1]が「0」でないということは、ビットストリーム中にmotion_residual[r][s][1]を表わすデータエレメントが存在し、動きベクトルの垂直方向成分が存在するということを意味しているので、その場合には、垂直方向成分のmotion_residual[r][s][1]を表わすデータエレメントが記述されている。

10

【0385】

次に、図65乃至図67を参照して、macroblock_typeについて説明する。macroblock_typeは、macroblock_quant、dct_type_flag、macroblock_motion_forward、及びmacroblock_motion_backwardなどのフラグから生成された可変長データである。macroblock_quantは、マクロブロックに対して量子化ステップサイズを設定するためのmacroblock_quantiser_scale_codeが設定されているか否かを示すフラグであって、ビットストリーム中にmacroblock_quantiser_scale_codeが存在する場合には、macroblock_quantは「1」の値を取る。

20

【0386】

dct_type_flagは、参照マクロブロックがフレームDCT又はフィールドDCTで符号化されているかを示すdct_typeが存在するか否かを示すためのフラグ（言い換えるとDCTされているか否かを示すフラグ）であって、ビットストリーム中にdct_typeが存在する場合には、このdct_type_flagは「1」の値を取る。macroblock_motion_forwardは、参照マクロブロックが前方予測されているか否かを示すフラグであって、前方予測されている場合には「1」の値を取る。macroblock_motion_backwardは、参照マクロブロックが後方予測されているか否かを示すフラグであって、後方予測されている場合には「1」の値を取る。

30

【0387】

なお、可変長フォーマットにおいては、伝送するビットレートを減少させるために、履歴情報を削減することができる。

【0388】

すなわち、macroblock_typeとmotion_vectors()は転送するが、quantiser_scale_codeを転送しない場合には、slice_quantiser_scale_codeを"00000"とすることで、ビットレートを減少させることができる。

40

【0389】

また、macroblock_typeのみ転送し、motion_vectors()、quantiser_scale_code、およびdct_typeを転送しない場合には、macroblock_typeとして、"not coded"を使用することで、ビットレートを減少させることができる。

【0390】

さらにまた、picture_coding_typeのみ転送し、slice()以下の情報は全て転送しない場合には、slice_start_codeを持たないpicture_data()を使用することで、ビットレートを減少させることができる。

【0391】

50

以上においては、user_data内の23ビットの連続する"0"が出ないようにする場合に、22ビット毎に"1"を挿入するようにしたが、22ビット毎でなくてもよい。また、連続する"0"の個数を数えて"1"を挿入するのではなく、Byte_alignを調べて挿入するようにすることも可能である。

【0392】

さらに、MPEGにおいては、23ビットの連続する"0"の発生を禁止しているが、実際には、バイトの先頭から23ビット連続する場合だけが問題とされ、バイトの先頭ではなく、途中から0が23ビット連続する場合は、問題とされない。従って、例えば24ビット毎に、LSB以外の位置に"1"を挿入するようにしてもよい。

【0393】

また、以上においては、履歴情報を、video elementary streamに近い形式にしたが、packetized elementary streamやtransport streamに近い形式にしてもよい。また、Elementary Streamのuser_dataの場所を、picture_dataの前としたが、他の場所にすることもできる。

【0394】

図15において説明したトランスコーダ101は、複数世代の符号化パラメータを履歴情報として後段のプロセスに提供するようにしたが、実際には、前述した履歴情報の全てが必要となるわけではない。例えば、比較的記憶容量に制限の無い大容量の記録媒体を備えた記録再生システムが、このトランスコーダの後段に接続されている場合には、符号化パラメータの中に前述した全ての履歴情報が記述されていても問題はないが、比較的記憶容量が小さい記録媒体を備えた記録再生装置が、このトランスコーダの後段に接続されている場合には、符号化ストリームのデータレートを少しでも低減するために、符号化パラメータの中に前述した全ての履歴情報を記述するのではなく、必要な履歴情報を記述する方が望ましい。また、他の例として、比較的伝送容量に制限の無い大容量の伝送容量を有した伝送路が、このトランスコーダの後段に接続されている場合には、符号化パラメータの中に前述した全ての履歴情報が記述されていても問題はないが、比較的伝送容量が少ない伝送路が、このトランスコーダの後段に接続されている場合には、符号化ストリームのデータレートを少しでも低減するために、符号化パラメータの中に前述した全ての履歴情報を記述するのではなく、必要な履歴情報を記述する方が望ましい。

【0395】

本発明の特徴的な点は、トランスコーディング装置の後段に接続される様々なアプリケーションのそれぞれに必要な履歴情報を、アプリケーションに応じて適応的にかつ選択的に、符号化ストリーム中に記述するものである。これを実現するために、本実施例では、符号化ストリーム中に、re-coding_stream_info という情報を記述している。

【0396】

以下に、図68を参照して、re-coding_stream_infoのシンタックス及びデータエレメントについて詳細に説明する。

【0397】

図68に示すように、re_coding_stream_info()関数は、user_data_start_code, re_coding_stream_info_ID, red_bw_flag, red_bw_indicator, marker_bit, num_other_bits, num_mv_bits, num_coef_bitsなどのデータエレメントより構成される。

【0398】

user_data_start_codeは、user_dataが開始することを表すスタートコードである。re_coding_stream_info_IDは、16ビットの整数であり、re_coding_stream_info()関数の識別のために用いられる。その値は、具体的には、"1001 0001 1110 1100" (0x91ec) とされる。

【0399】

red_bw_flagは、1ビットのフラグであり、全ての履歴情報に関するコーディングパラメータを送信する場合には0とされ、履歴情報に関するコーディングパラメータが選択的に送信する場合には1とされる。具体的には、図69に示すように、red_bw_flagが1で

10

20

30

40

50

ある場合、このフラグに続くred_bw_indicatorを調べることにより、5つのデータセットのうちどのデータセットを使用して履歴情報に関するコーディングパラメータが送られているのかを決定することができる。このデータセットは、history_stream()として、再符号化された符号化ストリームと一緒に伝送されるコーディングパラメータの組み合わせを決定するための情報である。したがって、このデータセットに従って、符号化ストリーム中に記述されるコーディングパラメータが選択される。

【0400】

red_bw_indicatorは、2ビットの整数であり、コーディングパラメータのデータセットを定義するためのインジケータである。具体的には、図69に示すように、red_bw_indicatorは、データセット2からデータセット5のうち、どのデータセットを表すかを示すデータである。

10

【0401】

よって、符号化ストリーム中に記述されている、このred_bw_flagとred_bw_indicatorとを参照することによって、5つのデータセットのうちどのデータセットを使用して履歴情報に関するコーディングパラメータが送られているのかを決定することができる。

【0402】

次に、図70を参照して、各データセットにおいて、履歴情報に関してどのようなコーディングパラメータが伝送されるかについて説明する。

【0403】

履歴情報は、大別すると、picture単位の情報と、macroblock単位の情報に分けることができる。slice単位の情報には、それに含まれるmacroblockの情報を収集することで得ることができ、GOP単位の情報には、それに含まれるpicture単位の情報を収集することで得ることができる。

20

【0404】

picture単位の情報には、1フレーム毎に1回伝送されるだけなので、符号化ストリーム中に挿入される履歴情報に対して占めるビットレートはそれほど大きくはない。これに対して、macroblock単位の情報には、各macroblock毎に伝送されるため、例えば1フレームの走査線数が525本で、フィールドレートが60フィールド/秒のビデオシステムにおいて、1フレームの画素数を720×480とすると、macroblock単位の情報には、1フレームあたり1350(=(720/16)×(480/16))回伝送することが必要となる。このため、履歴情報の相当の部分がmacroblock毎の情報で占められることになる。

30

【0405】

そこで、本実施例においては、符号化ストリーム中に挿入される履歴情報としては、少なくともpicture単位の情報には常に伝送するが、macroblock単位の情報には、アプリケーションに応じて選択して伝送するようにすることで、伝送する情報量を抑制することができる。

【0406】

図70に示したように、履歴情報として伝送されるmacroblock単位の情報には、例えばnum_coef_bits, num_mv_bits, num_other_bits, q_scale_code, q_scale_type, motion_type, mv_vert_field_sel[], mv[], mb_mfwd, mb_mbwd, mb_pattern, coded_block_pattern, mb_intra, slice_start, dct_type, mb_quant, skipped_mbなどがある。これらは、SMPTE-327Mにおいて定義されているmacroblock_rate_informationの要素を用いて表現されたものである。

40

【0407】

num_coef_bitsは、macroblockの符号量のうち、DCT係数に要した符号量を表す。num_mv_bitsは、macroblockの符号量のうち、動きベクトルに要した符号量を表す。num_other_bitsは、macroblockの符号量のうち、num_coef_bits及びnum_mv_bits以外の符号量を表す。

【0408】

q_scale_codeは、macroblockに適用されたq_scale_codeを表す。motion_typeは、macro

50

blockに適用された動きベクトルのtypeを表す。mv_vert_field_sel[][]は、macroblockに適用された動きベクトルのfield selectを表す。

【0409】

mv[][][]は、macroblockに適用された動きベクトルを表す。mb_mfwdは、macroblockの予測モードが前方向予測であることを示すフラグである。mb_mbwdは、macroblockの予測モードが後方向予測であることを示すフラグである。mb_patternは、macroblockのDCT係数の非0のものの有無を示すフラグである。

【0410】

coded_block_patternは、macroblockのDCT係数の非0のものの有無をDCTブロック毎に示すフラグである。mb_intraは、macroblockがintra_macroかそうでないかを示すフラグである。slice_startは、macroblockがsliceの先頭であるか否かを示すフラグである。dct_typeは、macroblockがfield_dctかframe_dctかを示すフラグである。

10

【0411】

mb_quantは、macroblockがquantiser_scale_codeを伝送するか否かを示すフラグである。skipped_mbは、macroblockがskipped macroblockであるか否かを示すフラグである。

【0412】

これらのコーディングパラメータは、常に全て必要であるわけではなく、トランスコーダの後段に接続されたアプリケーションに応じて、必要となるコーディングパラメータが変化する。例えば、num_coef_bitsやslice_startといったコーディングパラメータは、再エンコードした際のビットストリームをできる限り元の形に戻したいというtransparentという要求を有するアプリケーションにおいて必要となる。ここで、「transparentという要求」とは、入力ビットストリームに対して画質の劣化していない出力ビットストリームを生成することが可能になる要求である。

20

【0413】

つまり、トランスコーディング処理によって、ビットレートの変更のみを要求しているようなアプリケーションにおいては、これらのnum_coef_bitsやslice_startといったコーディングパラメータは必要ではない。また、非常に伝送路の制限が厳しい場合には、各ピクチャの符号化タイプのみを必要としているようなアプリケーションも存在する。

【0414】

このような状況を鑑みて、本実施例においては、履歴情報を伝送する際の、コーディングパラメータのデータセットの例として、例えば図70に示すようなデータセットを用意している。

30

【0415】

図70において、各データセットの中のコーディングパラメータに対応する値「2」は、その情報が履歴情報として符号化ストリーム中に存在し、利用可能であることを意味し、「0」は、その情報が符号化ストリーム中に存在しないことを意味する。「1」は、他の情報の存在を補助する目的のため、あるいは、シンタックス上存在するが、元のビットストリーム情報とは関係がないなど、その情報自身には意味がないことを表している。例えば、slice_startは、履歴情報を伝送する際のsliceの先頭のmacroblockにおいて、「1」になるが、本来のビットストリームに対して、sliceが必ずしも同一位置関係にあるわけではない場合には、履歴情報としては無意味になる。

40

【0416】

本実施例においては、選択されたデータセットに応じて、(num_coef_bits, num_mv_bits, num_other_bits), (q_scale_code, q_scale_type), (motion_type, mv_vert_field_sel[][]), (mb_mfwd, mb_mbwd), (mb_pattern), (coded_block_pattern), (mb_intra), (slice_start), (dct_type), (mb_quant), (skipped_mb)の各コーディングパラメータが、選択的に符号化ストリーム中に記述される。

【0417】

データセット1は、完全にtransparentなビットストリームを再構成することを目的としたデータセットである。このデータセット1によれば、入力ビットストリームに対して

50

画質劣化のほとんど無い出力ビットストリームを出力することができる精度の高いトランスコーディングが実現できる。データセット2も、完全にtransparentなビットストリームを再構成することを目的としたデータセットである。データセット3は、完全にtransparentなビットストリームを再構成することはできないが、視覚的にはほぼtransparentなビットストリームを再構成できるようにするためのデータセットである。データセット4は、transparentという観点からはデータセット3よりも劣るが、視覚上問題がないビットストリームの再構成ができるデータセットである。データセット5は、transparentという観点からはデータセット4よりも劣るが、少ない履歴情報でビットストリームの完全ではない再構成ができるデータセットである。

【0418】

10

これらのデータセットのうち、データセットの番号が小さいものほど、機能的には上位であるが、履歴情報を伝送するのに必要となる容量が多くなる。従って、想定するアプリケーションと履歴情報に使用できる容量を考慮することによって、伝送するデータセットが決定される。

【0419】

図70に示した5つのデータセットのうち、データセット1の場合、red_bw_flagは0とされ、データセット2乃至データセット5のとき、red_bw_flagは1とされる。これに対して、red_bw_indicatorは、データセット2の場合0とされ、データセット3の場合1とされ、データセット4の場合2とされ、データセット5の場合3とされる。

【0420】

20

従って、red_bw_indicatorは、red_bw_flagが1の場合に（データセット2乃至データセット5の場合に）規定される。

【0421】

さらに、red_bw_flagが0である場合（データセット1の場合）、マクロブロック毎に、marker_bit, num_other_bits, num_mv_bits, num_coef_bitsが記述される。これら4つのデータエレメントは、データセット2乃至データセット5の場合（red_bw_flagが1の場合）、符号化ストリーム中には記述されない。

【0422】

データセット5の場合、picture_data()関数（図59参照）を含めて、それ以下のシンタックス要素は伝送されない。つまり、picture_data()関数に含まれる複数のslice()関数（図60参照）に関する符号化パラメータは全く伝送されない。よって、データセット5の場合、選択された履歴情報は、picture_typeなどのpicture単位のコーディングパラメータのみの伝送を意図したものとなる。

30

【0423】

データセット1乃至データセット4の場合、picture_data()関数に含まれる複数のslice()関数に関する符号化パラメータが存在する。しかしながら、このslice()関数によって決定されるsliceのアドレス情報と、元のビットストリームのsliceのアドレス情報は、選択されたデータセットに依存する。データセット1またはデータセット2の場合、履歴情報の元となったビットストリームのsliceのアドレス情報と、slice()関数によって決定されるsliceのアドレス情報とは、同一である必要がある。

40

【0424】

macroblock()関数のシンタックス要素（図61参照）は、選択されたデータセットに依存する。macroblock_escape, macroblock_address_increment, macroblock_modes()関数は、常に符号化ストリーム中に存在する。しかしながら、macroblock_escapeとmacroblock_address_incrementの情報としての有効性は、選択されたデータセットによって決定される。履歴情報に関するコーディングパラメータのデータセットが、データセット1またはデータセット2の場合、元のビットストリームのskipped_mb情報と同じものが伝送される必要がある。

【0425】

データセット4の場合、motion_vectors()関数は符号化ストリーム中には存在しない。

50

データセット 1 乃至データセット 3 の場合、macroblock_modes() 関数のmacroblock_type によって、motion_vectors() 関数の符号化ストリーム中における存在が決定される。データセット 3 またはデータセット 4 の場合には、coded_block_pattern() 関数は符号化ストリーム中には存在しない。データセット 1 とデータセット 2 の場合、macroblock_modes() 関数のmacroblock_typeによって、coded_block_pattern() 関数の符号化ストリーム中における存在が決定される。

【0 4 2 6】

macroblock_modes() 関数のシンタックス要素（図 6 2 参照）は、選択されたデータセットに依存する。macroblock_typeは、常に存在する。データセット 4 の場合、flame_motion_type, field_motion_type, dct_typeは符号化ストリーム中には存在しない。

10

【0 4 2 7】

macroblock_typeより得られるパラメータの情報としての有効性は、選択されたデータセットによって決定される。

【0 4 2 8】

データセット 1 またはデータセット 2 の場合、macroblock_quantは、元のビットストリームのmacroblock_quantと同じである必要がある。データセット 3 またはデータセット 4 の場合、macroblock_quantは、macroblock() 関数内のquantiser_scale_codeの存在を表し、元のビットストリームと同じである必要はない。

【0 4 2 9】

データセット 1 乃至データセット 3 の場合、macroblock_motion_forwardとmacroblock_motion_backwardは、元のビットストリームと同一である必要がある。データセットがデータセット 4 またはデータセット 5 である場合、その必要はない。

20

【0 4 3 0】

データセット 1 またはデータセット 2 の場合、macroblock_patternは、元のビットストリームと同一である必要がある。データセット 3 の場合、macroblock_patternは、dct_typeの存在を示すのに用いられる。データセット 4 の場合、データセット 1 乃至データセット 3 における場合のような関係は成立しない。

【0 4 3 1】

履歴情報のコーディングパラメータのデータセットがデータセット 1 乃至データセット 3 の場合、macroblock_intraは、元のビットストリームと同一である必要がある。データセット 4 の場合には、その限りでない。

30

【0 4 3 2】

次に、データセットに関する情報を含んでいる符号化ストリームに対するトランスコーダ 1 0 1 の処理及び、設定されたデータセットに基づいて、符号化ストリームを生成するトランスコーダ 1 0 1 の処理について、図 1 5 に示したトランスコーダ 1 0 1 を参照して説明する。

【0 4 3 3】

この図 1 5 に示したトランスコーダの例は、第 3 世代のエンコード処理によって生成された符号化ストリーム S T (3 r d) を受け取り、この符号化ストリーム S T (3 r d) の G O P 構造又は／及びビットレートを変換した新たな符号化ストリーム S T (4 t h) を生成するための例である。

40

【0 4 3 4】

まず最初に、復号装置 1 0 2 は、第 3 世代の符号化ストリーム S T (3 r d) を符号化する際に使用した第 3 世代のコーディングパラメータ (3 t h) を、第 3 世代の符号化ストリーム S T (3 r d) から抽出し、この抽出した符号化パラメータに基づいて、符号化ストリーム S T (3 r d) をデコードし、ベースバンドビデオ信号を生成する。さらに、復号装置 1 0 2 は、抽出した第 3 世代のコーディングパラメータ (3 t h) を、ヒストリ情報多重化装置 1 0 3 に出力する。さらに、復号装置 1 0 2 は、第 3 世代の符号化ストリーム S T (3 r d) のピクチャレイヤからuser_data()を抽出して、そのuser_data()ヒストリデコーディング装置 1 0 4 に供給する。

50

【0435】

ヒストリデコーディング装置104は、復号装置102から供給されたuser_data()からhistry_stream()を抽出する。このhistry_stream()は可変長符号化されているデータエレメントからなるストリームであるので、ヒストリデコーディング装置104は、histry_stream()に対して可変長復号化処理を施す。その結果、それぞれが所定のデータ長を有したデータエレメントからなるストリームを生成することができる。次に、ヒストリデコーディング装置104は、可変長復号化したデータストリームのシンタックスをパーズング(parsing)。尚、パーズングとは、ストリームの構文を解釈することである。

【0436】

このパーズング処理を行う際、ヒストリデコーディング装置104は、history_stream()関数内の re_coding_stream_info() 関数の中に記述されているred_bw_flag と red_bw_indicator を参照する。ヒストリデコーディング装置104は、ストリーム中から抜き出したred_bw_flag と red_bw_indicator を参照することによって、受け取ったhistry_stream()に対して、5つのデータセットのうち、どのデータセットが設定されているかを判断する。これによって、ヒストリデコーディング装置104は、red_bw_flag と red_bw_indicatorによってデータセットの種類を決定することによって、どのコーディングパラメータがhistry_stream()に含まれているかを知ることができる。

【0437】

具体的には、red_bw_flag = 0 の場合には、データセット1であるので、histry_stream()中には、picture_data()関数として、num_coef_bits, num_mv_bits, num_other_bits, q_scale_code, q_scale_type, motion_type, mv_vert_field_sel[][], mv[][][], mb_mfwd, mb_mbwd, mb_pattern, coded_block_pattern, mb_intra, slice_start, dct_type, mb_quant, skipped_mb 等の全てのコーディングパラメータが記述されている。

【0438】

red_bw_flag = 1 かつ red_bw_indicator = 0 の場合には、データセット2であるので、histry_stream()中には、picture_data()関数として、q_scale_code, q_scale_type, motion_type, mv_vert_field_sel[][], mv[][][], mb_mfwd, mb_mbwd, mb_pattern, coded_block_pattern, mb_intra, slice_start, dct_type, mb_quant, skipped_mb 等のコーディングパラメータが記述されている。

【0439】

red_bw_flag = 1 かつ red_bw_indicator = 1 の場合には、データセット3であるので、histry_stream()中には、picture_data()関数として、q_scale_code, q_scale_type, motion_type, mv_vert_field_sel[][], mv[][][], mb_mfwd, mb_mbwd, mb_pattern, mb_intra, slice_start, dct_type, mb_quant, skipped_mb 等のコーディングパラメータが記述されている。

【0440】

red_bw_flag = 1 かつ red_bw_indicator = 2 の場合には、データセット4であるので、histry_stream()中には、picture_data()関数として、q_scale_code, q_scale_type のコーディングパラメータが記述されている。

【0441】

red_bw_flag = 1 かつ red_bw_indicator = 3 の場合には、データセット5であるので、histry_stream()中には、picture_data()関数として、どのコーディングパラメータも記述されていない。

【0442】

ヒストリデコーディング装置104は、red_bw_flag と red_bw_indicatorの情報を参照することによって、histry_stream()中に含まれているコーディングパラメータを、histry_stream()中から抽出する。図15に示したトランスコーダーの実施例においては、このヒストリデコーディング装置104トランスコーダーに供給された入力符号化ストリームは、第3世代の符号化処理によって生成された符号化ストリームであるので、から出力される履歴情報は、第1世代のコーディングパラメータと第2世代のコーディングパラ

10

20

30

40

50

メータである。

【0443】

ヒストリ情報多重化装置103は、復号装置102から供給される第3世代の符号化パラメータ(3th)と、ヒストリデコーディング装置104から供給される過去のコーディングパラメータ(1st, 2nd)とを、復号装置102から供給されるベースバンドのビデオデータに、図68から図73に示したようなフォーマットに従って多重化する。

【0444】

ヒストリ情報分離装置105は、ヒストリ情報多重化装置103より供給されたベースバンドビデオデータを受け取り、このベースバンドビデオデータから第1、第2及び第3世代のコーディングパラメータ(1st、2nd、3rd)を抽出し、符号化装置106

10

【0445】

符号化装置106は、ヒストリ情報分離装置105からベースバンドビデオデータとコーディングパラメータ(1st、2nd、3rd)を受け取り、受け取ったコーディングパラメータに基づいてベースバンドビデオデータを再符号化する。この際、符号化装置106は、過去の符号化処理において生成されたコーディングパラメータ(1st、2nd、3rd)と、供給されたベースバンドビデオデータから新たに生成されたコーディングパラメータとから、符号化処理に最適なコーディングパラメータを選択する。符号化装置106は、供給されたベースバンドビデオデータから新たに生成されたコーディングパラメータを使用して符号化処理を行う場合には、符号化装置106は、前述した「通常符号化処理」を行い、符号化装置106は、過去のコーディングパラメータ(1st、2nd、3rd)のうちからいずれかのコーディングパラメータを再利用する場合には、前述した「パラメータ再利用符号化処理」を行う。

20

【0446】

ネットワーク上に設けられたコンピュータ100は、復号化装置102のデコーディング処理及び符号化装置106のエンコーディング処理をコントロールするための装置である。例えば、符号化装置106から出力される符号化ストリームを伝送するための伝送路の容量を検出し、その伝送容量に応じて適切なデータセットを、前述した5つのデータセットの中から選択する。また、符号化装置106の出力段に接続されている装置の記憶容量を検出し、その記憶容量に応じて適切なデータセットを、前述した5つのデータセット

30

【0447】

符号化装置106は、コンピュータ100からデータセットを示す情報を受け取り、それに基づいてred_bw_flag及びred_bw_indicatorを生成する。コンピュータ100から与えられた情報が、データセット1の場合にはred_bw_flag = 0となり、データセット2の場合には、red_bw_flag = 1かつred_bw_indicator = 0となり、データセット3の場合には、red_bw_flag = 1かつred_bw_indicator = 1となり、データセット4の場合には、red_bw_flag = 1かつred_bw_indicator = 2となり、データセット5の場合には、red_bw_flag = 1かつred_bw_indicator = 3となる。

【0448】

40

符号化装置106は、決定されたred_bw_flagの値及びred_bw_indicatorの値に応じて、histry_stream()として、符号化ストリーム中に記述するコーディングパラメータを選択し、選択したコーディングパラメータをhistry_stream()として符号化ストリーム中に記述するとともに、red_bw_flag及びred_bw_indicatorをre_coding_stream_info()として符号化ストリーム中に記述する。尚、histry_stream()として伝送されるコーディングパラメータの選択処理は、第1世代、第2世代及び第3世代の過去のコーディングパラメータに対して、夫々行われる処理である。

【0449】

符号化装置106は、red_bw_flag = 0の場合には、histry_stream()中には、picture_data()関数として、num_coef_bits, num_mv_bits, num_other_bits, q_scale_code, q_

50

scale_type, motion_type, mv_vert_field_sel[][], mv[][][], mb_mfwd, mb_mbwd, mb_pattern, coded_block_pattern, mb_intra, slice_start, dct_type, mb_quant, skipped_mb 等の全てのコーディングパラメータを符号化ストリーム中に記述する。

【0450】

符号化装置106は、red_bw_flag = 1 かつ red_bw_indicator = 0 の場合には、history_stream()中には、picture_data()関数として、q_scale_code, q_scale_type, motion_type, mv_vert_field_sel[][], mv[][][], mb_mfwd, mb_mbwd, mb_pattern, coded_block_pattern, mb_intra, slice_start, dct_type, mb_quant, skipped_mb のコーディングパラメータを符号化ストリーム中に記述する。

【0451】

符号化装置106は、red_bw_flag = 1 かつ red_bw_indicator = 1 の場合には、history_stream()中には、picture_data()関数として、q_scale_code, q_scale_type, motion_type, mv_vert_field_sel[][], mv[][][], mb_mfwd, mb_mbwd, mb_pattern, mb_intra, slice_start, dct_type, mb_quant, skipped_mb のコーディングパラメータを符号化ストリーム中に記述する。

【0452】

符号化装置106は、red_bw_flag = 1 かつ red_bw_indicator = 2 の場合には、history_stream()中には、picture_data()関数として、q_scale_code, q_scale_typeのコーディングパラメータを符号化ストリーム中に記述する。

【0453】

符号化装置106は、red_bw_flag = 1 かつ red_bw_indicator = 3 の場合には、history_stream()中には、picture_data()関数として、どのコーディングパラメータも符号化ストリーム中に記述しない。

【0454】

すなわち、符号化装置106は、供給された過去のコーディングパラメータの全てをhistory_stream()として伝送するのではなく、コンピュータ100から指定されたデータセットを示す情報に基づいて、history_stream()として伝送されるコーディングパラメータを選択する。従って、符号化装置105は、受け取ったデータセットを示す情報に基づいて、様々なデータ容量のhistory_stream()を含んだ符号化ストリームを生成することができる。また、符号化装置105は、符号化装置105の後段に接続される伝送メディアの伝送容量、記録メディアの記憶容量又はアプリケーションに見合った適切なデータ量からなるhistory_stream()を含んだ符号化ストリームを生成することができる。また、本実施例のトランスコーダによれば、このようにhistory_stream()として伝送する符号化パラメータを、符号化装置の後段に接続されるアプリケーションに応じて、選択するようにしているので、アプリケーションに応じた履歴を最適なデータ量で伝送するようにすることができる。

【0455】

次に、図71から図74を参照して、これらの履歴情報を、ヒストリ情報多重化装置103が出力するベースバンドビデオ信号に多重して伝送するフォーマットについて説明する。

【0456】

図71は、ベースバンドビデオ信号に多重して伝送するフォーマット「Re_Coding information Bus macroblock format」を示す図である。このマクロブロックは、16×16 (= 256) ビットで構成される。そして、そのうちの図71において上から3行目と4行目に示す32ビットが、picrate_elementとされる。このpicrate_elementには、図72乃至図74に示すPicture rate elementsが記述される。図72の上から2行目に1ビットのred_bw_flagが規定されており、また、3行目に3ビットのred_bw_indicatorが規定されている。即ち、これらのフラグred_bw_flag, red_bw_indicatorは、図71のpicrate_elementとして伝送される。

【0457】

図71のその他のデータについて説明すると、SRIB_sync_codeは、このフォーマットのマクロブロックの最初の行が左詰めにアライメントされていることを表すコードであり、具体的には、“11111”に設定される。fr_fl_SRIBは、picture_structureがフレームピクチャ構造の場合（その値が“11”である場合）、1に設定され、Re_Coding Information Bus macroblockが16ラインを超えて伝送されることを表し、picture_structureがフレーム構造ではない場合、0に設定され、Re_Coding Information Busが16ラインを超えて伝送されることを意味する。この機構により、Re_Coding Information Busが、空間的かつ時間的にデコードされたビデオフレームまたはフィールドの対応する画素にロックされる。

【0458】

10

SRIB_top_field_firstは、元のビットストリームに保持されているtop_field_firstと同じ値に設定され、関連するビデオのRe_Coding Information Busの時間的アライメントをrepeat_first_fieldとともに表している。SRIB_repeat_first_fieldは、元のビットストリームに保持されているrepeat_first_fieldと同じ値に設定される。first fieldのRe_Coding Information Busの内容は、このフラグに示されるように繰り返される必要がある。

【0459】

422_420_chromaは、元のビットストリームが4:2:2または4:2:0のいずれであるかを表す。その値の0は、ビットストリームが4:2:0であり、色差信号のアップサンプリングが、4:2:2のビデオが出力されるように行われたことを表す。その値の0

20

【0460】

rolling_SRIB_mb_refは、16ビットのモジュロ65521を表し、この値は、毎マクロブロック毎にインクリメントされる。この値は、フレームピクチャ構造のフレームに渡って連続している必要がある。さもなくば、この値は、フィールドに渡って連続している必要がある。この値は、0から65520の間の所定の値に初期化される。これにより、レコーダのシステムに、ユニークなRe_Coding Information Busの識別子を組み込むことが許容される。

【0461】

Re_Coding Information Bus macroblockのその他のデータの意味は、上述した通りであるので、ここでは省略する。

30

【0462】

図75に示すように、図71の256ビットのRe_Coding Information Busのデータは、1ビットずつ、色差データのLSBであるCb[0][0], Cr[0][0], Cb[1][0], Cr[1][0]に配置される。図75に示すフォーマットにより、4ビットのデータを送ることができるので、図71の256ビットのデータは、図75のフォーマットを64（＝256／4）個送ることで伝送することができる。

【0463】

本発明のトランスコーダによれば、過去の符号化処理において生成された符号化パラメータを、現在の符号化処理において再利用するようにしているので、復号処理及び符号化処理を繰り返したとしても画質劣化が発生しない。つまり、復号処理及び符号化処理の繰り返しによる画質劣化の蓄積を低減することができる。

40

【0464】

図76と図77は、本発明のトランスコーダをビデオテープレコーダに適用した場合の構成例を表している。図76は、ビデオテープレコーダ601の記録系の構成例を表しており、図77は、ビデオテープレコーダ601の再生系の構成例を表している。

【0465】

図76のビデオテープレコーダ601は、トランスコーダ101R、チャンネルエンコーディング装置602、記録ヘッド603により構成されている。トランスコーダ101Rの構成は、図37に示したトランスコーダと基本的に同様とされている。この構成例に

50

においては、トランスコーダ101Rにおいては、Long GOPのビットストリームSTが、Short GOPのビットストリームSTに変換される。

【0466】

トランスコーダ101Rの符号化装置106より出力された、第4世代の符号化ストリームSTが、チャンネルエンコーディング装置602に供給される。上述したように、この第4世代の符号化ストリームSTのピクチャ層のユーザデータエリアには、第1世代乃至第3世代の符号化パラメータを含むユーザデータuser_dataが記録されている。

【0467】

チャンネルエンコーディング装置602は、入力された第4世代の符号化ストリームに、誤り訂正のためのパリティ符号を付けた後、例えば、NRZI変調方式でチャンネルエンコードし、記録ヘッド603に供給する。記録ヘッド603は入力された符号化ストリームを磁気テープ604に記録する。

【0468】

図77に示すように、再生系においては、磁気テープ604から、再生ヘッド611により信号が生成され、チャンネルデコーディング装置612に供給される。チャンネルデコーディング装置612は、再生ヘッド611から供給された信号をチャンネルデコードし、パリティを用いて、誤り訂正する。

【0469】

チャンネルデコーディング装置612により出力された第4世代の符号化ストリームSTは、トランスコーダ101Pに入力される。トランスコーダ101Pの基本的な構成は、図37に示したトランスコーダと同様の構成とされている。

【0470】

トランスコーダ101Pの復号装置102は、第4世代の符号化ストリームから、第1世代乃至第3世代の符号化パラメータを含むユーザデータuser_dataを抽出し、ヒストリデコーディング装置104と、符号化装置106に供給する。ヒストリデコーディング装置104は、入力されたユーザデータuser_dataを復号し、得られた第1世代乃至第3の符号化パラメータを符号化装置106に供給する。

【0471】

復号装置102はまた、第4世代の符号化ストリームSTを復号し、ベースバンドビデオ信号と、第4世代の符号化パラメータを出力する。ベースバンドビデオ信号は、符号化装置106に供給され、第4世代の符号化パラメータは、符号化装置106と、ヒストリエンコーディング装置107に供給される。

【0472】

ヒストリエンコーディング装置107は、入力された第4世代の符号化パラメータを、ユーザデータuser_dataに変換し、符号化装置106に供給する。

【0473】

上述したように、符号化装置106のコントローラ70は、オペレータによって指定されたGOP構造から決定された各ピクチャのピクチャタイプと、ヒストリ情報（ユーザデータuser_data）に含まれるピクチャタイプが一致するか否かを判断する。そしてその判断結果に対応して、上述した、「通常符号化処理」、または「パラメータ再利用符号化処理」を実行する。この処理により符号化装置106から、Short GOPからLong GOPに変換された、第4世代の符号化ストリームSTが出力される。この符号化ストリームSTのユーザデータuser_dataには、第1世代乃至第4世代の符号化パラメータがヒストリ情報として記録されている。

【0474】

図76と図77に示したビデオテープレコーダ601においては、ヒストリ情報をピクチャレイヤのuser_dataに記録するようにしたが、ヒストリ情報は、磁気テープ604のビデオデータとは異なる領域に記録することも可能である。図78と図79は、この場合のビデオテープレコーダ601の構成例を表している。図78は、ビデオテープレコーダ601の記録系の構成例を表しており、図79は、再生系の構成例を表している。

【0475】

図78に示すように、このビデオテープレコーダ601においては、そのトランスコーダ101Rの復号装置102より出力されたユーザデータuser_dataが、ヒストリデコーディング装置104に入力され、そこで過去の符号化パラメータ（この例の場合、第1世代と第2世代の符号化パラメータ）が復号され、符号化装置106に供給されている。また、この例においては、磁気テープ604にユーザデータuser_dataとして、ヒストリ情報を記録する必要がないので、図15に示したヒストリエンコーディング装置107のうち、ヒストリVLC211のみが採用されている。そしてこのヒストリVLC211に、復号装置102が出力した符号化パラメータ（この例の場合、第3世代の符号化パラメータ）と、ヒストリデコーディング装置104がユーザデータuser_dataから復号、出力した符号化パラメータ（この例の場合、第1世代と第2世代の符号化パラメータ）が供給されている。ヒストリVLC211は、この第1世代乃至第3世代の符号化パラメータを可変長符号化し、図40乃至図46、または図47に示したhistory_streamを生成し、マルチプレクサ621に供給する。

10

【0476】

マルチプレクサ621には、また、符号化装置106より出力された第4世代の符号化ストリームSTが入力されている。マルチプレクサ621は、符号化装置106より供給された符号化ストリーム（ビットストリーム）をヒストリVLC211より供給されたヒストリよりも安全な領域に多重化する。

20

【0477】

例えば、図80に示すように、磁気テープ604において、符号化装置106より出力されたビデオストリームは、シンクコードに近い位置に記録され、ヒストリVLC211より出力されたhistory_streamは、シンクコードからビデオストリームより、より離れた位置に記録される。特殊再生時などにおいて、ビデオストリームを検索するとき、最初にシンクコードが検出され、そのシンクコードを基準として、それに続くビデオストリームが検索される。従って、シンクコードに近い位置にビデオストリームを配置した方が、高速再生時などにおいても、より確実にビデオデータを再生することが可能になる。history_streamは、高速再生時などにおいて、必要とされる情報ではない。そこで、このhistory_streamは、シンクコードからより離れた位置に配置しても、それほど支障は生じない。

30

【0478】

マルチプレクサ621により多重化された信号は、チャンネルエンコーディング装置602に入力され、チャンネルエンコードされた後、記録ヘッド603により、磁気テープ604に記録される。

【0479】

このように、この例においては、history_streamが、ビデオデータとは異なる位置に多重化されるため、仮にそこにスタートコードが現れたとしても、ビデオデータとは充分区別することが可能である。そこで、この例においては、マーカビットを挿入し、history_streamをconverted_history_streamとする必要はない。

【0480】

また、符号化パラメータをhistory_streamのフォーマットにしないで、そのままマルチプレクサ621に供給し、多重化させることも可能データあるが、そのようにすると、圧縮されていないので、符号化パラメータのデータ量が多くなり、磁気テープ604の利用効率が低下する。そこで、ヒストリVLC211により圧縮し、history_streamのフォーマットにして、多重化するようにするのが好ましい。

40

【0481】

図79に示すように、ビデオテープレコーダ601の再生系においては、磁気テープ604から、再生ヘッド611により再生された信号が、チャンネルデコーディング装置612でチャンネルデコードされる。デマルチプレクサ631は、チャンネルデコーディング装置612でチャンネルデコードされる。デマルチプレクサ631は、チャンネルデコーディング装置612から供給された第4世代の符号化ストリームSTを、ビデオストリー

50

ムと、history_streamとに分離し、ビデオストリームを復号装置102に供給し、history_streamをヒストリVLD203に供給する。

【0482】

すなわちこの例においては、図15に示したヒストリデコーディング装置104のうち、ヒストリVLD203のみが採用される。

【0483】

ヒストリVLD203は、history_streamを可変長復号処理し、得られた第1世代乃至第3世代の符号化パラメータを符号化装置106に出力する。

【0484】

また、デマルチプレクサ631より出力されたhistory_streamは、コンバータ212' 10
入力される。このコンバータ212'と、後段のユーザデータフォーマッタ213'は、
ヒストリエンコーディング装置107に内蔵されているコンバータ212、およびユーザ
データフォーマッタ213（図15参照）とは、別個のものであるが、それらと同一の機
能を果たすものである。

【0485】

すなわちコンバータ212'は、デマルチプレクサ631より入力されたhistory_stream
にマーカビットを付加して、converted_history_streamを生成し、ユーザデータフォー
マッタ213'に出力する。ユーザデータフォーマッタ213'は、入力されたconverte
d_history_streamをuser_dataに変換し、符号化装置106に出力する。このuser_dataに
は、第1世代乃至第3世代の符号化パラメータが含まれていることになる。 20

【0486】

復号装置102は、デマルチプレクサ631から入力されたビデオストリームを復号し
、ベースバンドビデオ信号を符号化装置106に出力する。また、復号装置102は、
第4世代の符号化パラメータを符号化装置106に供給するとともに、ヒストリ円コー
ディング装置107に出力する。ヒストリエンコーディング装置107は、入力された第4
世代の符号化パラメータからuser_dataを生成し、符号化装置106に出力する。

【0487】

符号化装置106は、図77における符号化装置106と同様に、「通常符号化処理」
または「パラメータ再利用符号化処理」を実行し、第5世代の符号化ストリームSTを出力
する。この第5世代の符号化ストリームSTには、そのピクチャ層のuser_dataに、第1世
代乃至第4世代の符号化パラメータが記録されている。 30

【0488】

以上の説明から理解できるように、本発明のトランスコーダによれば、過去の符号化処
理において生成された符号化パラメータを、現在の符号化処理において生成された符号化
ストリームのユーザデータエリアに記述するようにし、生成されたビットストリームは、
MPEG規格に準じた符号化ストリームであるので、既存のどのデコーダでも復号処理を行
うことができる。さらには、本発明のトランスコーダによれば、過去の符号化処理におけ
る符号化パラメータを伝送するために専用線のようなものを設ける必要がないので、従来
のデータストリーム伝送環境をそのまま使用して、過去の符号化パラメータを伝送するこ
とができる。 40

【0489】

本実施例のトランスコーダによれば、過去の符号化処理において生成された符号化パラ
メータを、選択的に現在の符号化処理において生成された符号化ストリーム中に記述す
るようにしているので、出力されるビットストリームのビットレートを極端に上げるこ
となく、過去の符号化パラメータを伝送することができる。

【0490】

本実施例のトランスコーダによれば、過去の符号化パラメータと現在の符号化ラメータ
の中から、現在の符号化処理に最適な符号化パラメータを選択して符号化処理を行うよ
うにしているので、復号処理及び符号化処理を繰り返したとしても、画質劣化が蓄積さ
れることはない。 50

【0491】

本実施例のトランスコーダによれば、過去の符号化パラメータの中から、ピクチャタイプに応じて現在の符号化処理に最適な符号化パラメータを選択して符号化処理を行うようにしているので、復号処理及び符号化処理を繰り返したとしても、画質劣化が蓄積されることはない。

【0492】

また、本実施例のトランスコーダによれば、過去の符号化パラメータに含まれるピクチャタイプに基づいて、過去の符号化パラメータを再利用するか否かを決定しているので、最適な符号化処理を行うことができる。なお、上記各処理を行うコンピュータプログラムは、磁気ディスク、光ディスク、光磁気ディスク、半導体メモリなどの記録媒体に記録して提供するほか、インターネット、ATM、デジタル衛星などのネットワークを介して伝送し、ユーザの記録媒体に記録させることで提供することができる。

10

【0493】

また、本実施例のトランスコーダによれば、トランスコーダによって再符号化された符号化ストリームに対して、それまでの符号化処理における符号化履歴の組み合わせを表すデータセットに関する情報を記述するようにしたので、少ない容量の伝送路を介して伝送可能な履歴情報を含むストリームを生成することが可能となる。

【0494】

本実施例のトランスコーダによれば、過去の符号化処理において使用された複数の符号化パラメータを、選択的に組み合わせて、その符号化履歴情報を生成し、符号化ストリームに重畳するようにしたので、再符号化に伴う画像の劣化を抑制可能なストリームを、少ない容量のメディアを介して伝送することが可能となる。

20

【0495】

本実施例のトランスコーダによれば、データセットを示す情報に基づいて、符号化パラメータを抽出し、抽出した符号化パラメータに基づいて、再符号化された符号化ストリームを生成するようにしたので、再符号化に伴う画像の劣化を抑制した、少ない伝送容量の伝送メディアに伝送可能なストリームを符号化することが可能となる。

【0496】

また、本実施例のトランスコーダによれば、検出された過去の符号化処理における符号化履歴を記録媒体に記録するようにしたので、符号化ストリームを記録媒体に記録した場合においても、画質の劣化を抑制することが可能となる。

30

【0497】

本実施例のトランスコーダによれば、記録媒体から再生された符号化ストリームに含まれる符号化履歴を検出し、再符号化された符号化ストリームと多重化して出力するようにしたので、記録媒体から再生された符号化ストリームを再度トランスコーディングする場合においても、画質の劣化を抑制することが可能となる。

【産業上の利用可能性】

【0498】

本発明は、コーディングシステム及び方法、符号化装置および方法、復号化装置及び方法、記録装置及び方法、ならびに再生装置及び方法に関し、特に、MPEG規格に基づいて過去に符号化処理が施されたことがある符号化ストリームに対して再符号化処理を施すことによって、異なるGOP (Group of Pictures) 構造や異なるビットレートを有した再符号化ストリームを生成するためのトランスコーダなどに利用できる。

40

【図面の簡単な説明】

【0499】

【図1】図1は、高効率符号化の原理を説明する図である。

【図2】図2は、画像データを圧縮する場合におけるピクチャタイプを説明する図である。

。

【図3】図3は、画像データを圧縮する場合におけるピクチャタイプを説明する図である。

。

50

- 【図 4】図 4 は、動画像信号を符号化する原理を説明する図である。
- 【図 5】図 5 は、動画像信号を符号化し、復号する装置の構成を示すブロック図である。
- 【図 6】図 6 は、画像データの構成を説明する図である。
- 【図 7】図 7 は、図 5 のエンコーダ 1 8 の構成を示すブロック図である。
- 【図 8】図 8 は、図 7 の予測モード切替回路 5 2 の動作を説明する図である。
- 【図 9】図 9 は、図 7 の予測モード切替回路 5 2 の動作を説明する図である。
- 【図 10】図 10 は、図 7 の予測モード切替回路 5 2 の動作を説明する図である。
- 【図 11】図 11 は、図 7 の予測モード切替回路 5 2 の動作を説明する図である。
- 【図 12】図 12 は、図 5 のデコーダ 3 1 の構成を示すブロック図である。
- 【図 13】図 13 は、ピクチャタイプに対応したSNR制御を説明する図である。 10
- 【図 14】図 14 は、本発明を適用したトランスコーダ 1 0 1 の構成を示すブロック図である。
- 【図 15】図 15 は、図 14 のトランスコーダ 1 0 1 のより詳細な構成を示すブロック図である。
- 【図 16】図 16 は、図 14 の復号装置 1 0 2 に内蔵される復号化装置 1 0 2 の構成を示すブロック図である。
- 【図 17】図 17 は、マクロブロックの画素を説明する図である。
- 【図 18】図 18 は、符号化パラメータが記録される領域を説明する図である。
- 【図 19】図 19 は、図 14 の符号化装置 1 0 6 に内蔵される符号化装置 1 0 6 の構成を示すブロック図である。 20
- 【図 20】図 20 は、図 15 のヒストリVLC 2 1 1 の構成例を示すブロック図である。
- 【図 21】図 21 は、図 15 のヒストリVLD 2 0 3 の構成例を示すブロック図である。
- 【図 22】図 22 は、図 15 のコンバータ 2 1 2 の構成例を示すブロック図である。
- 【図 23】図 23 は、図 22 のスタッフ回路 3 2 3 の構成例を示すブロック図である。
- 【図 24】図 24 は、図 22 のコンバータ 2 1 2 の動作を説明するタイミングチャートである。
- 【図 25】図 25 は、図 15 のコンバータ 2 0 2 の構成例を示すブロック図である。
- 【図 26】図 26 は、図 25 のディリット回路 3 4 3 の構成例を示すブロック図である。
- 【図 27】図 27 は、図 15 のコンバータ 2 1 2 の他の構成例を示すブロック図である。
- 【図 28】図 28 は、図 15 のコンバータ 2 0 2 の他の構成例を示すブロック図である。 30
- 【図 29】図 29 は、図 15 のユーザデータフォーマッタ 2 1 3 の構成例を示すブロック図である。
- 【図 30】図 30 は、図 14 のトランスコーダ 1 0 1 が実際に使用される状態を示す図である。
- 【図 31】図 31 は、符号化パラメータが記録される領域を説明する図である。
- 【図 32】図 32 は、図 14 の符号化装置 1 0 6 の変更可能ピクチャタイプ判定処理を説明するフローチャートである。
- 【図 33】図 33 は、ピクチャタイプが変更される例を示す図である。
- 【図 34】図 34 は、ピクチャタイプが変更される他の例を示す図である。
- 【図 35】図 35 は、図 14 の符号化装置 1 0 6 の量子化制御処理を説明する図である。 40
- 【図 36】図 36 は、図 14 の符号化装置 1 0 6 の量子化制御処理を説明するフローチャートである。
- 【図 37】図 37 は、密結合されたトランスコーダ 1 0 1 の構成を示すブロック図である。
- 【図 38】図 38 は、ビデオシーケンスのストリームのシンタックスを説明する図である。
- 【図 39】図 39 は、図 38 のシンタックスの構成を説明する図である。
- 【図 40】図 40 は、固定長の履歴情報を記録するhistory_stream()のシンタックスを説明する図である。
- 【図 41】図 41 は、固定長の履歴情報を記録するhistory_stream()のシンタックスを説 50

明する図である。

【図 4 2】図 4 2 は、固定長の履歴情報を記録するhistory_stream()のシンタックスを説明する図である。

【図 4 3】図 4 3 は、固定長の履歴情報を記録するhistory_stream()のシンタックスを説明する図である。

【図 4 4】図 4 4 は、固定長の履歴情報を記録するhistory_stream()のシンタックスを説明する図である。

【図 4 5】図 4 5 は、固定長の履歴情報を記録するhistory_stream()のシンタックスを説明する図である。

【図 4 6】図 4 6 は、固定長の履歴情報を記録するhistory_stream()のシンタックスを説明する図である。 10

【図 4 7】図 4 7 は、可変長の履歴情報を記録するhistory_stream()のシンタックスを説明する図である。

【図 4 8】図 4 8 は、sequence_header()のシンタックスを説明する図である。

【図 4 9】図 4 9 は、sequence_extension()のシンタックスを説明する図である。

【図 5 0】図 5 0 は、extension_and_user_data()のシンタックスを説明する図である。

【図 5 1】図 5 1 は、user_data()のシンタックスを説明する図である。

【図 5 2】図 5 2 は、group_of_pictures_header()のシンタックスを説明する図である。

【図 5 3】図 5 3 は、picture_header()のシンタックスを説明する図である。

【図 5 4】図 5 4 は、picture_coding_extension()のシンタックスを説明する図である。 20

【図 5 5】図 5 5 は、extension_data()のシンタックスを説明する図である。

【図 5 6】図 5 6 は、quant_matrix_extension()のシンタックスを説明する図である。

【図 5 7】図 5 7 は、copyright_extension()のシンタックスを説明する図である。

【図 5 8】図 5 8 は、picture_display_extension()のシンタックスを説明する図である。

。

【図 5 9】図 5 9 は、picture_data()のシンタックスを説明する図である。

【図 6 0】図 6 0 は、slice()のシンタックスを説明する図である。

【図 6 1】図 6 1 は、macroblock()のシンタックスを説明する図である。

【図 6 2】図 6 2 は、macroblock_modes()のシンタックスを説明する図である。

【図 6 3】図 6 3 は、motion_vectors(s)のシンタックスを説明する図である。 30

【図 6 4】図 6 4 は、motion_vector(r,s)のシンタックスを説明する図である。

【図 6 5】図 6 5 は、I ピクチャに対するmacroblock_typeの可変長符号を説明する図である。

【図 6 6】図 6 6 は、P ピクチャに対するmacroblock_typeの可変長符号を説明する図である。

【図 6 7】図 6 7 は、B ピクチャに対するmacroblock_typeの可変長符号を説明する図である。

【図 6 8】図 6 8 は、re_coding_stream_info()のシンタックスを説明する図である。

【図 6 9】図 6 9 は、red_bw_flag, red_bw_indicatorを説明する図である。

【図 7 0】図 7 0 は、履歴情報のコーディングパラメータのデータセットを説明する図である。 40

【図 7 1】図 7 1 は、Re_Coding Information Bus macroblock formationを説明する図である。

【図 7 2】図 7 2 は、Picture rate elementsを説明する図である。

【図 7 3】図 7 3 は、Picture rate elementsを説明する図である。

【図 7 4】図 7 4 は、Picture rate elementsを説明する図である。

【図 7 5】図 7 5 は、Re_Coding Information Busが記録される領域を説明する図である。

。

【図 7 6】図 7 6 は、ビデオテープレコーダの記録系の構成例を表すブロック図である。

【図 7 7】図 7 7 は、ビデオテープレコーダの再生系の構成例を表すブロック図である。 50

【図 78】図 78 は、ビデオテープレコーダの記録系の他の構成例を表すブロック図である。

【図 79】図 79 は、ビデオテープレコーダの再生系の他の構成例を表すブロック図である。

【図 80】図 80 は、ビデオストリームとhistory_streamの記録位置を説明する図である。

【符号の説明】

【0500】

1、106……符号化装置、2、102……復号装置、3……記録媒体、14、33……フレームメモリ、17、32……フォーマット変換回路、18……エンコーダ、19……記録回路、30……再生回路、31……デコーダ、50……動きベクトル検出回路、57……量子化回路、58……可変長符号化回路、59……送信バッファ、64、87……動き補償回路、70……コントローラ、81……受信バッファ、82……可変長復号回路、83……逆量子化回路、84……IDCT回路、100……コンピュータ、101……トランスコーダ、103……ヒストリ情報多重化装置、104……ヒストリデコーディング装置、105……ヒストリ情報分離装置、107……ヒストリエンコーディング装置、201……ユーザデータデコーダ、202、212……コンバータ、203、211……ヒストリVLC。

10

【図 1】

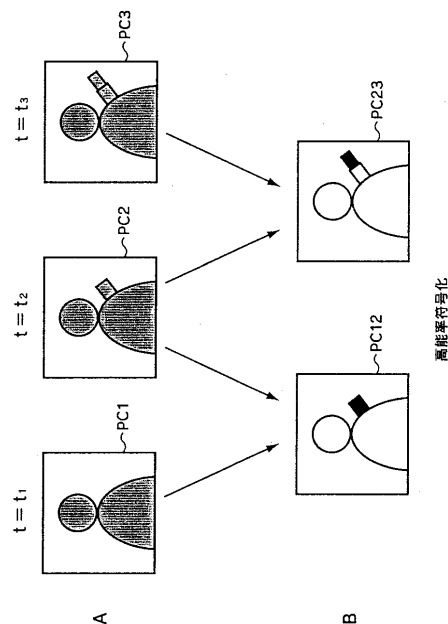


図 1

【図 2】

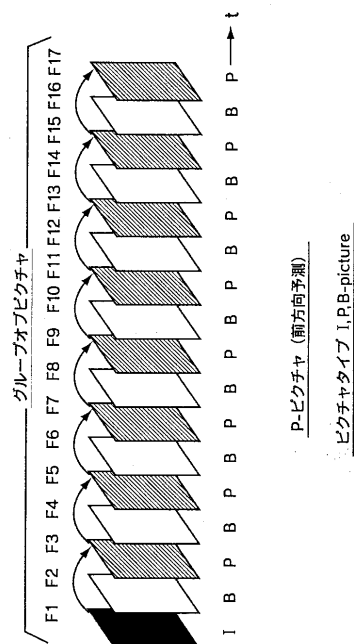


図 2

【図 3】

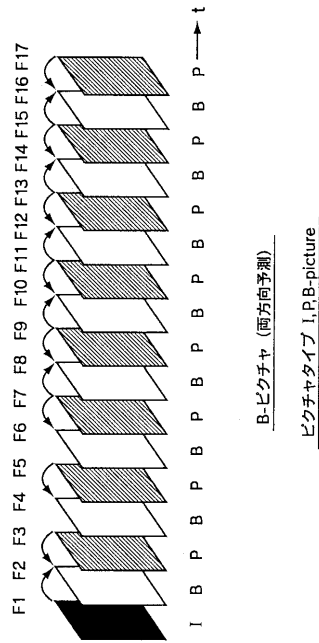


図 3

【図 4】

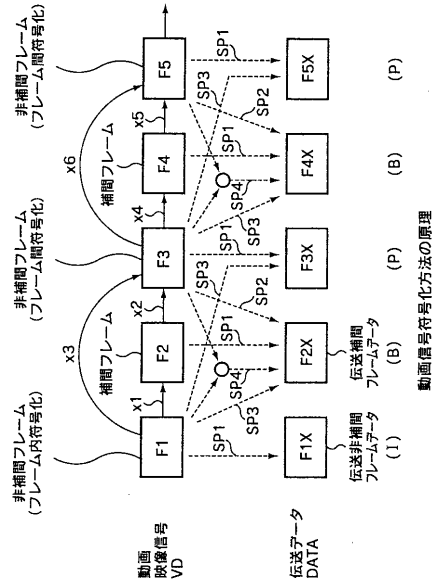


図 4

【図 5】

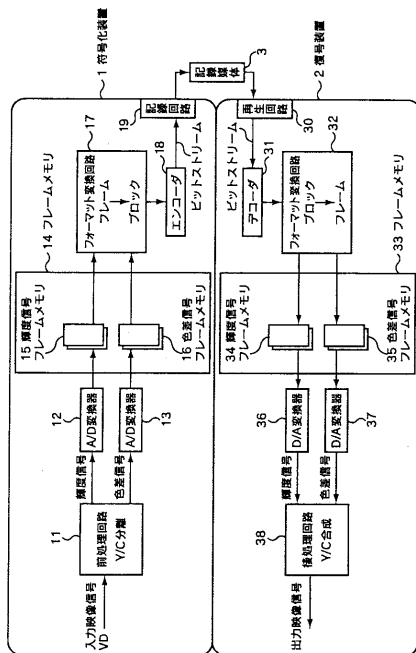


図 5

【図 6】

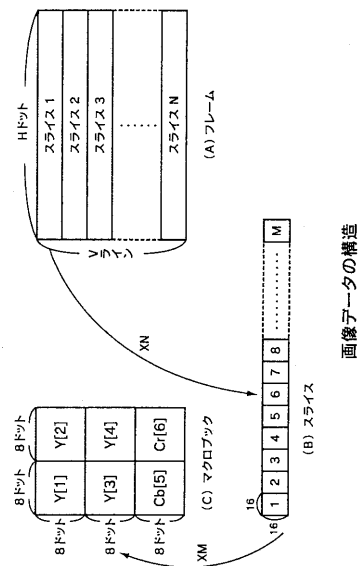
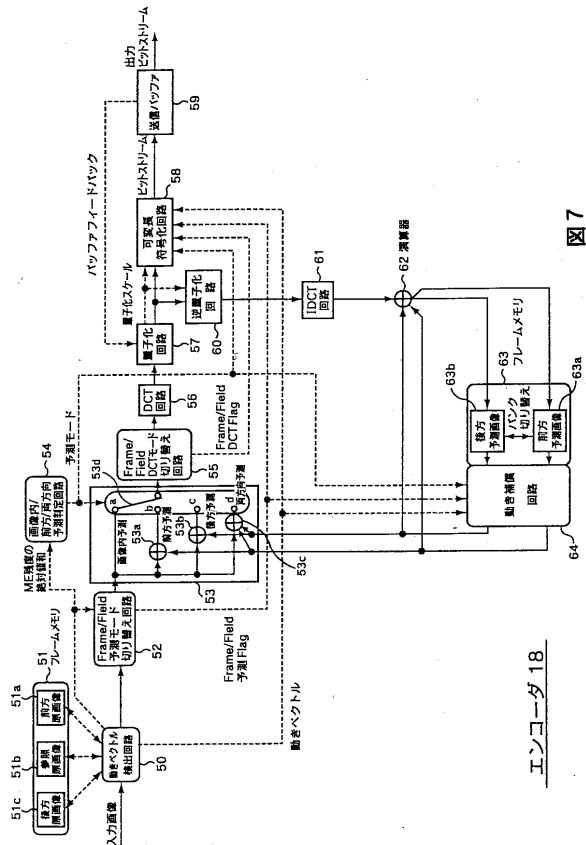


図 6

【図 7】



【图 8】

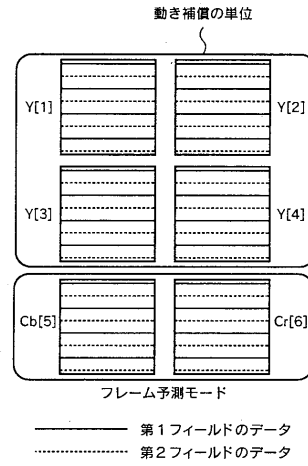


图 8

【图 9】

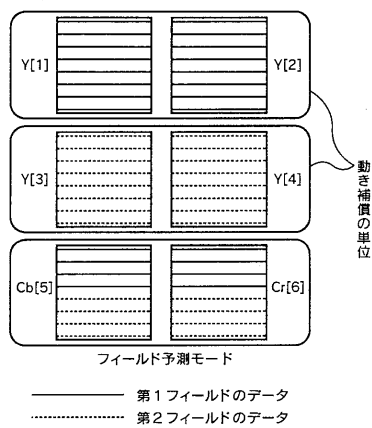


图 9

【図 10】

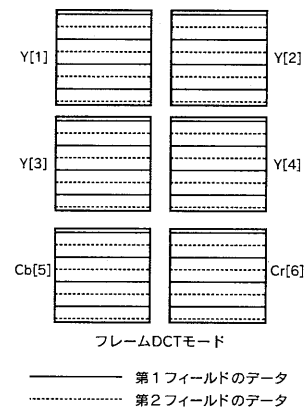


图 10

【図 1 1】

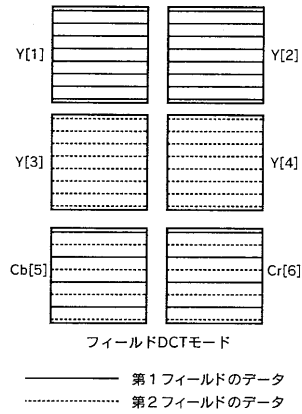


図 1 1

【図 1 2】

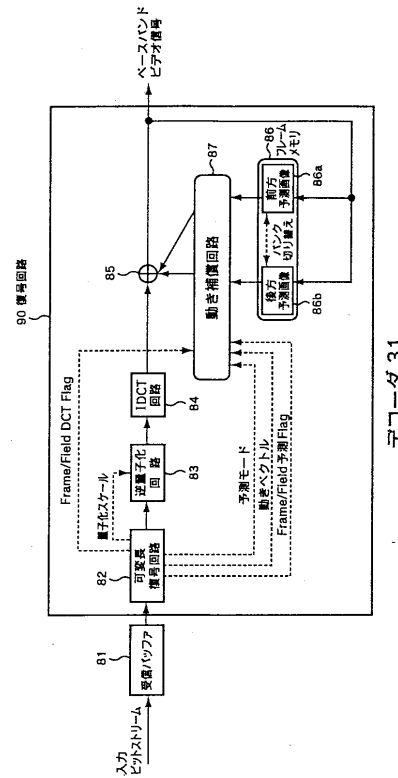


図 1 2

【図 1 3】

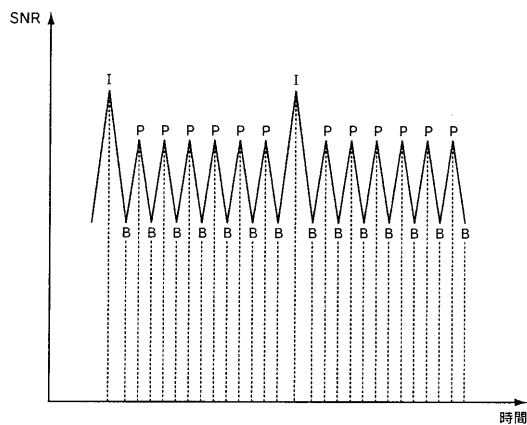


図 1 3

【図 1 4】

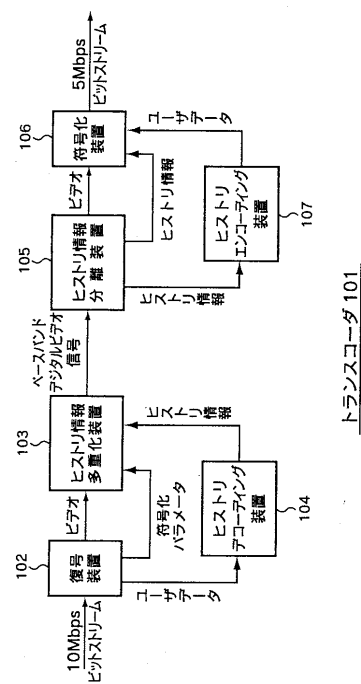


図 1 4

【図 15】

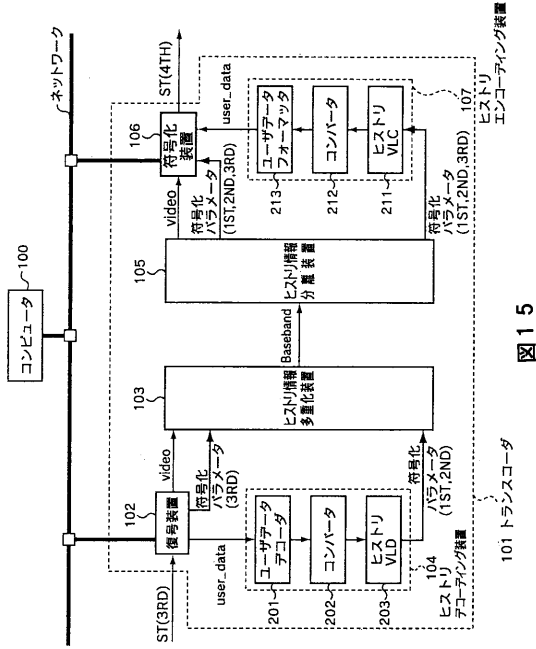


図 15

【図 16】

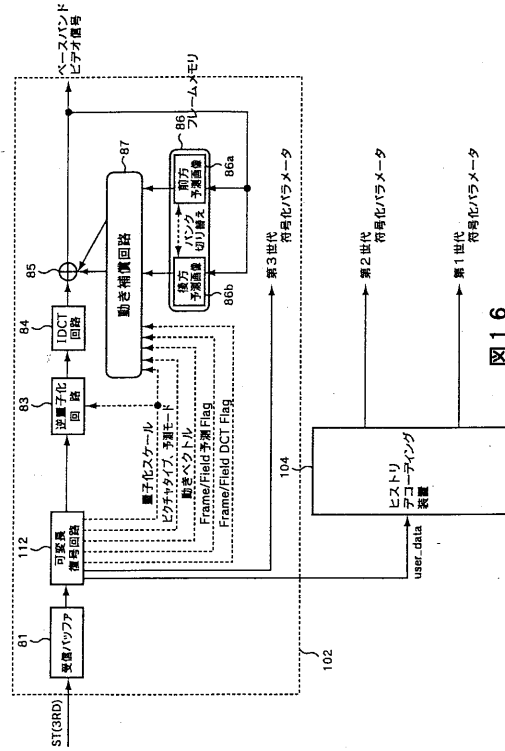


図 16

【図 17】

0	1	2	...	13	14	15
16	17	18	...	29	30	31
32	33	34	...	45	46	47
...	...	...	...	...	...	...
208	209	210	...	221	222	223
224	225	226	...	237	238	239
240	241	242	...	251	254	255

マクロブロック

図 17

【図 18】

画像データ領域												ヒストリ情報領域											
D9	Cb[0][9]	Y[0][9]	Cr[0][9]	Y[1][9]	Y[2][9]	Cb[1][9]	Y[3][9]	Cr[1][9]	Y[4][9]	Y[5][9]	Y[6][9]	D8	Cb[0][8]	Y[0][8]	Cr[0][8]	Y[1][8]	Y[2][8]	Cb[1][8]	Y[3][8]	Cr[1][8]	Y[4][8]	Y[5][8]	Y[6][8]
D7	Cb[0][7]	Y[0][7]	Cr[0][7]	Y[1][7]	Y[2][7]	Cb[1][7]	Y[3][7]	Cr[1][7]	Y[4][7]	Y[5][7]	Y[6][7]	D6	Cb[0][6]	Y[0][6]	Cr[0][6]	Y[1][6]	Y[2][6]	Cb[1][6]	Y[3][6]	Cr[1][6]	Y[4][6]	Y[5][6]	Y[6][6]
D5	Cb[0][5]	Y[0][5]	Cr[0][5]	Y[1][5]	Y[2][5]	Cb[1][5]	Y[3][5]	Cr[1][5]	Y[4][5]	Y[5][5]	Y[6][5]	D4	Cb[0][4]	Y[0][4]	Cr[0][4]	Y[1][4]	Y[2][4]	Cb[1][4]	Y[3][4]	Cr[1][4]	Y[4][4]	Y[5][4]	Y[6][4]
D3	Cb[0][3]	Y[0][3]	Cr[0][3]	Y[1][3]	Y[2][3]	Cb[1][3]	Y[3][3]	Cr[1][3]	Y[4][3]	Y[5][3]	Y[6][3]	D2	Cb[0][2]	Y[0][2]	Cr[0][2]	Y[1][2]	Y[2][2]	Cb[1][2]	Y[3][2]	Cr[1][2]	Y[4][2]	Y[5][2]	Y[6][2]
D1	Cb[0][1]	Y[0][1]	Cr[0][1]	Y[1][1]	Y[2][1]	Cb[1][1]	Y[3][1]	Cr[1][1]	Y[4][1]	Y[5][1]	Y[6][1]	D0	Cb[0][0]	Y[0][0]	Cr[0][0]	Y[1][0]	Y[2][0]	Cb[1][0]	Y[3][0]	Cr[1][0]	Y[4][0]	Y[5][0]	Y[6][0]

図 18

【図 19】

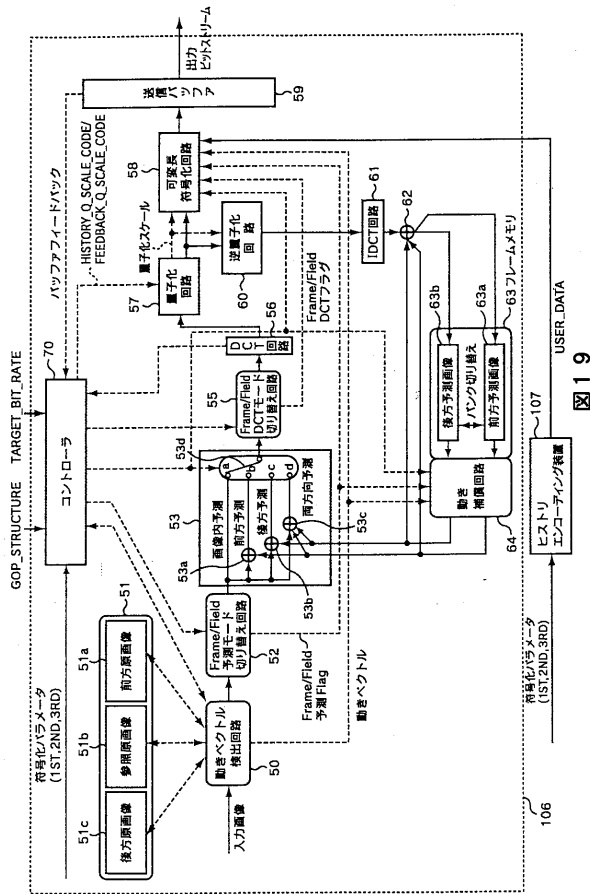
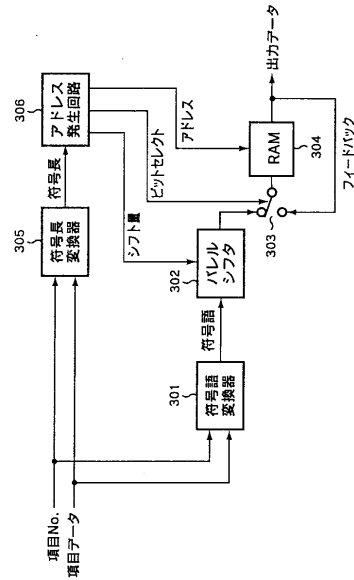


図 19

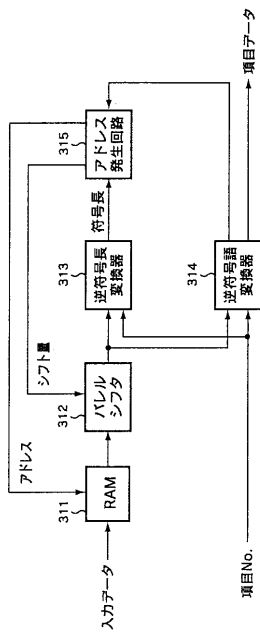
【図 20】



ヒストリ VLC 211

図 20

【図 21】



ヒストリ VLD 203

図 21

【図 22】

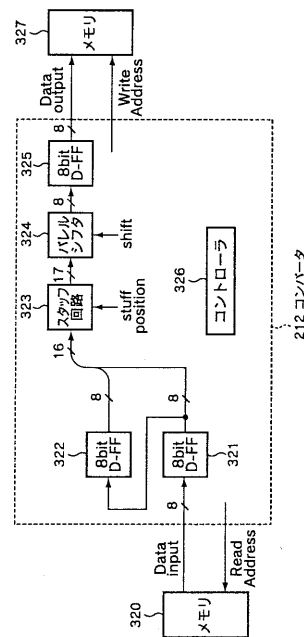


図 22

【図 2 3】

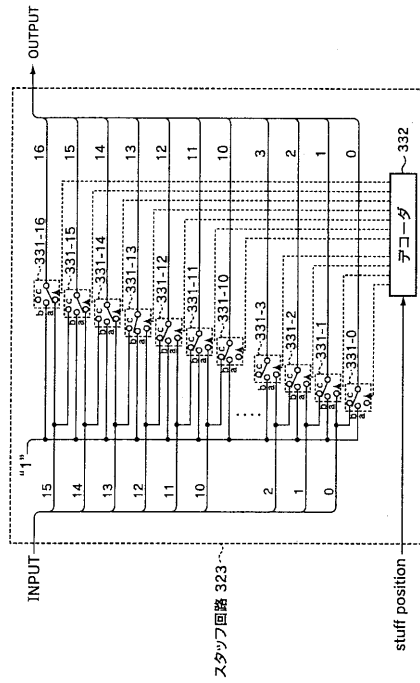


図 2 3

【図 2 4】

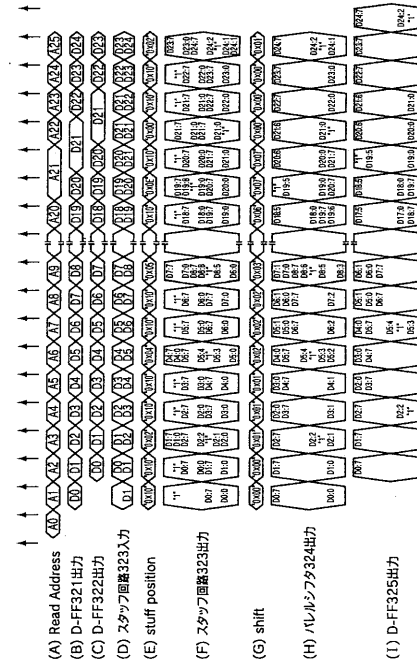


図 2 4

【図 2 5】

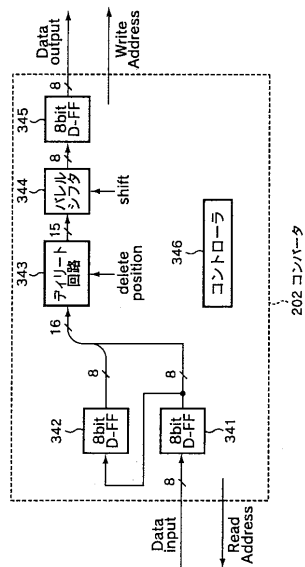


図 2 5

【図 2 6】

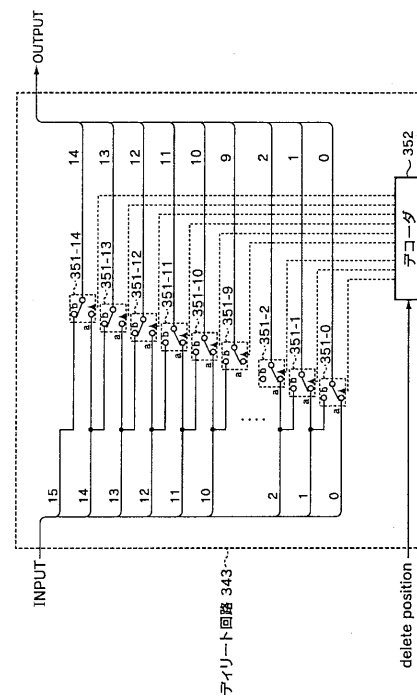


図 2 6

【図 27】

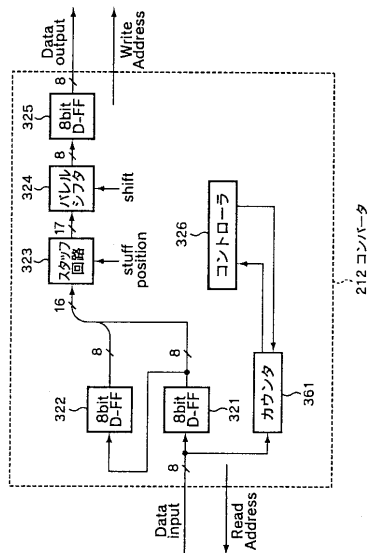


図 27

【図 28】

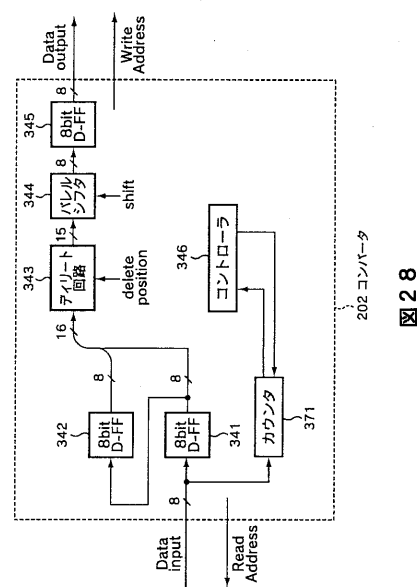


図 28

【図 29】

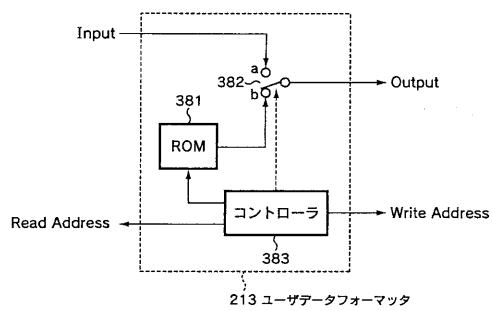


図 29

【図 30】

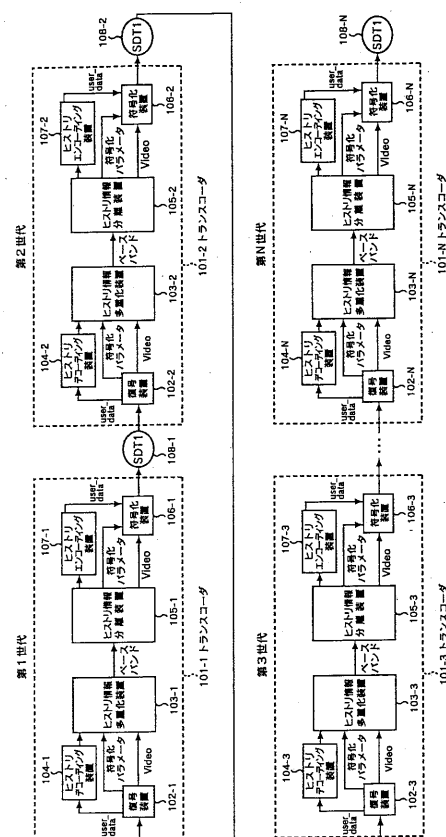
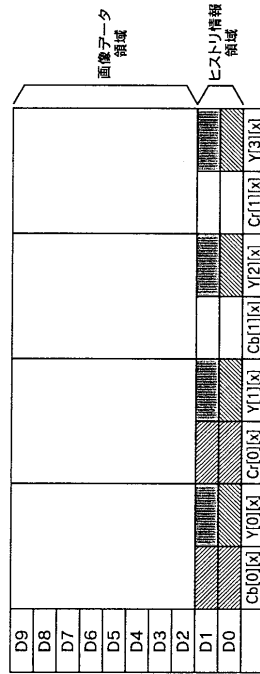
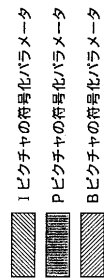


図 30

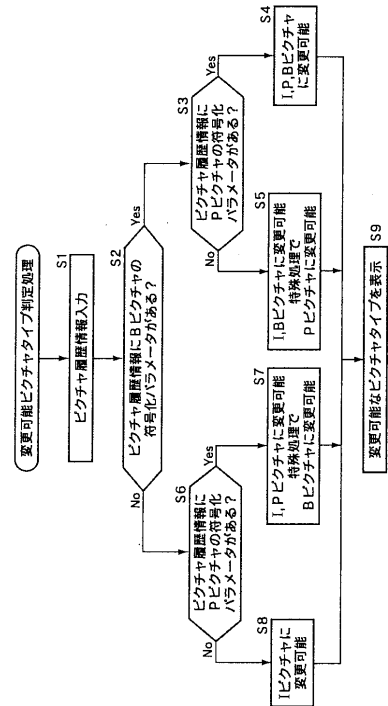
【図 3 1】



31

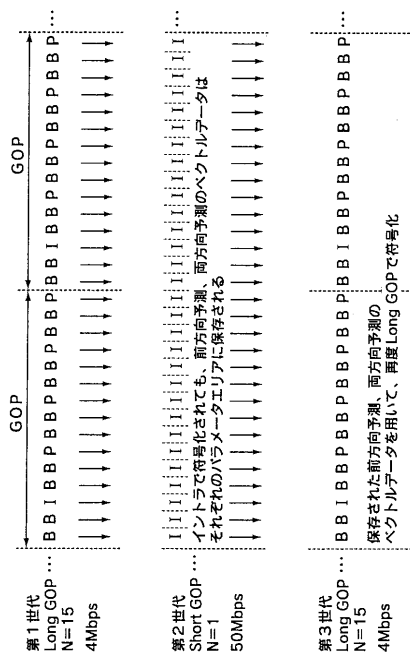


【図 3 2】



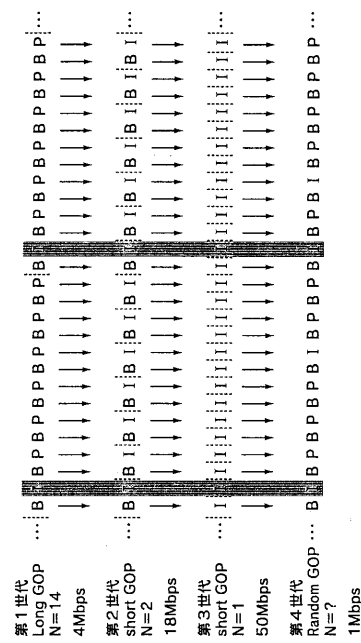
32

【図 3 3】



3
3
☒

【図 3 4】



34

【図 35】

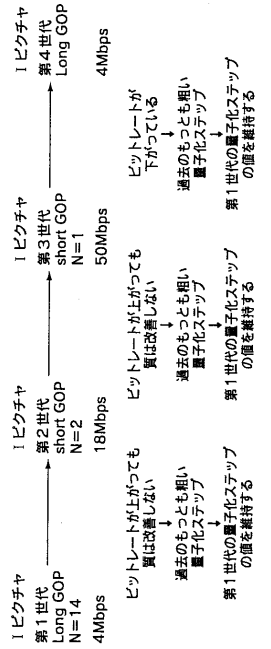


図 35

【図 36】

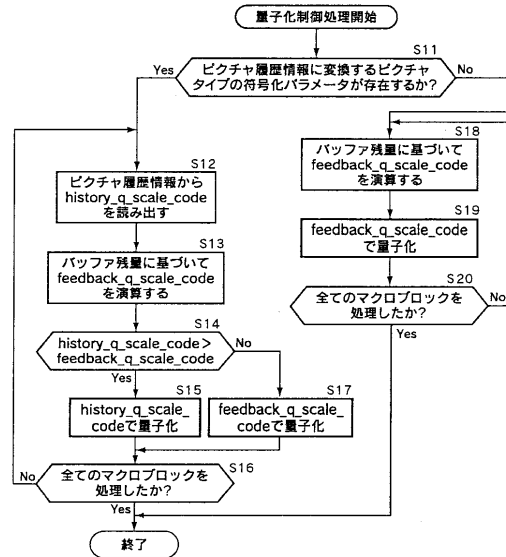


図 36

【図 37】

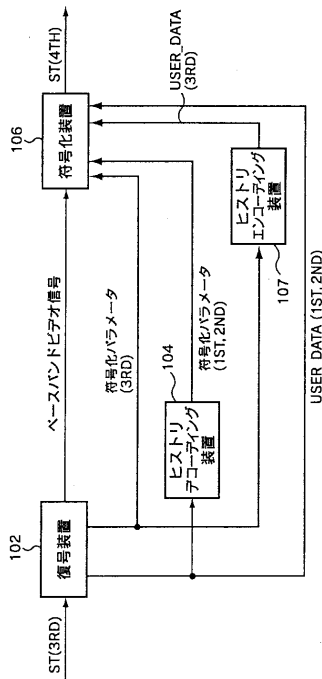


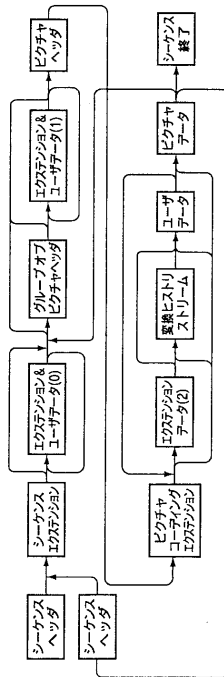
図 37

【図 38】

stream with history data		
	No. of bits	Mnemonic
video_sequence () {		
next_start_code ()		
sequence_header ()		
sequence_extension ()		
do {		
extension_and_user_data (0)		
do {		
if(nextbits() == group_start_code){		
group_of_pictures_header(1)		
extension_and_user_data (1)		
}		
picture_header()		
picture_coding_extension()		
while((nextbits() == extension_start_code)		
(nextbits() == user_data_start_code)){		
if(nextbits() == extension_start_code){		
extension_data(2)		
if(nextbits() == user_data_start_code){		
user_data_start_code	32	bslbf
if(nextbits() == History_Data_ID){		
History_Data_ID	8	bslbf
converted_history_stream()		
}		
else{		
user_data()		
}		
}		
picture_data()		
}while((nextbits() == picture_start_code)		
(nextbits() == group_start_code))		
if(nextbits() != sequence_end_code){		
sequence_header ()		
sequence_extension ()		
}		
}while(nextbits() != sequence_end_code)		
sequence_end_code		
}	32	bslbf

図 38

【図 3 9】



93X

【図 40】

history stream(40-1)		
history_stream(){	bits	value
sequence_header		
sequence_header_code	32	000001B3
sequence_header_present_flag	1	
horizontal_size_value	12	
marker_bit	1	1
vertical_size_value	12	
aspect_ratio_information	4	
frame_rate_code	4	
marker_bit	1	1
bit_rate_value	18	
marker_bit	1	1
vbv_buffer_size_value	10	
constrained_parameter_flag	1	0
load_intra_quantiser_matrix	1	
load_non_intra_quantiser_matrix	1	
marker_bits	5	1F
intra_quantiser_matrix[64]	8*64	
non_intra_quantiser_matrix[64]	8*64	
sequence_extension		
extension_start_code	32	000001B5
extension_start_code_identifier	4	1
sequence_extension_present_flag	1	
profile_and_level_indication	8	
progressive_sequence	1	
chroma_format	2	
horizontal_size_extension	2	
vertical_size_extension	2	
marker_bit	1	1
bit_rate_extension	12	
vbv_buffer_size_extension	8	
low_delay	1	
marker_bit	1	1

図 40

【図 4 1】

	bits	value
frame_rate_extension_n	2	
frame_rate_extension_d	5	
marker_bits	6	3F
sequence_display_extension		
extension_start_code	32	000001B5
extension_start_code_identifier	4	2
sequence_display_extension_present_flag	1	
video_format	3	
colour_description	1	
colour_primaries	8	
transfer_characteristics	8	
marker_bit	1	1
matrix_coefficients	8	
display_horizontal_size	14	
marker_bit	1	1
display_vertical_size	14	
marker_bit	1	1
macroblock_assignment_in_user_data		
macroblock_assignment_present_flag	1	
marker_bit	7	7F
v_phase	8	
h_phase	8	
group_of_picture_header		
group_start_code	32	000001B8
group_of_picture_header_present_flag	1	
time_code	25	
closed_gop	1	
broken_link	1	
marker_bits	4	F
picture_header		
picture_start_code	32	00000100

图 4-1

【図 4 2】

	bits	value
temporal_reference	10	
picture_coding_type	3	
marker_bit	1	1
vbv_delay	16	
full_pel_forward_vector	1	
forward_f_code	3	
full_pel_backward_vector	1	
marker_bit	1	1
backward_f_code	3	
marker_bit	1	1
picture_coding_extension		
extension_start_code	32	000001B5
extension_start_code_identifier	4	8
f_code[0][0]	4	
f_code[0][1]	4	
f_code[1][0]	4	
f_code[1][1]	4	
intra_dc_precision	2	
picture_structure	2	
top_field_first	1	
frame_pred_frame_dct	1	
concealment_motion_vectors	1	
q_scale_type	1	
marker_bit	1	1
intra_vlc_format	1	
alternate_scan	1	
repeat_first_field	1	
chroma_420_type	1	
progressive_frame	1	
composite_display_flag	1	
v_axis	1	
field_sequence	3	
sub_carrier	1	
burst_amplitude	7	

42

【図 4 3】

history stream(40-4)		
	bits	value
marker_bits	1	1
sub_carrier_phase	8	
quant_matrix_extension		
extension_start_code	32	000001B5
extension_start_code_identifier	4	3
quant_matrix_extension_present_flag	1	
load_intra_quantiser_matrix	1	
marker_bits	2	3
intra_quantiser_matrix[64]	8*64	
load_non_intra_quantiser_matrix	1	
marker_bits	7	7F
non_intra_quantiser_matrix[64]	8*64	
load_chroma_intra_quantiser_matrix	1	
marker_bits	7	7F
chroma_intra_quantiser_matrix[64]	8*64	
load_chroma_non_intra_quantiser_matrix	1	
marker_bits	7	7F
chroma_non_intra_quantiser_matrix[64]	8*64	
copyright_extension		
extension_start_code	32	000001B5
extension_start_code_identifier	4	4
copyright_extension_present_flag	1	
copyright_flag	1	
copyright_identifier	8	
original_or_copy	1	
marker_bit	1	
copyright_number_1	20	
marker_bit	1	
copyright_number_2	22	
marker_bit	1	
copyright_number_3	22	3F
marker_bits	6	

図 4 3

【図 4 4】

history stream(40-5)		
	bits	value
picture_display_extension		
extension_start_code	32	000001B5
extension_start_code_identifier	4	7
picture_display_extension_present_flag	1	
frame_centre_horizontal_offset_1	16	
marker_bit	1	1
frame_centre_vertical_offset_1	16	
marker_bit	1	1
frame_centre_horizontal_offset_2	16	
marker_bit	1	1
frame_centre_vertical_offset_2	16	
marker_bit	1	1
frame_centre_horizontal_offset_3	16	
marker_bit	1	1
frame_centre_vertical_offset_3	16	
marker_bit	6	3F
re_coding_stream_information		
user_data_start_code	32	000001B2
re_coding_stream_info_ID	16	91EC
red_bw_flag	1	
red_bw_indicator	2	
marker_bit	5	1F
user_data		
user_data_start_code	32	000001B2
user_data	2048	
while(macroblock i=macroblock_count){		
macroblock		
macroblock_address_h	8	
macroblock_address_v	8	
slice_header_present_flag	1	
skipped_macroblock_flag	1	
marker_bit	1	1
macroblock_modes()		
macroblock_quant	1	
macroblock_motion_forward	1	
macroblock_motion_backward	1	
macroblock_pattern	1	
macroblock_intra	1	

図 4 4

【図 4 5】

history stream(40-6)		
	bits	value
spatial_temporal_weight_code_flag	1	
frame_motion_type	2	
field_motion_type	2	
dct_type	1	
marker_bits	2	3
quantiser_scale_code	5	
marker_bits	3	7
PMV[0][0][0]	14	
marker_bits	2	3
PMV[0][0][1]	14	
motion_vertical_field_select[0][0]	1	
marker_bit	1	1
PMV[0][1][0]	14	
marker_bits	2	3
PMV[0][1][1]	14	
motion_vertical_field_select[0][1]	1	
marker_bit	1	1
PMV[1][0][0]	14	
marker_bits	2	3
PMV[1][0][1]	14	
motion_vertical_field_select[1][0]	1	
marker_bit	1	1
PMV[1][1][0]	14	
marker_bits	2	3
PMV[1][1][1]	14	
motion_vertical_field_select[1][1]	1	
marker_bit	1	1
coded_block_pattern	12	
marker_bits	4	F
num_mv_bits	8	
num_coef_bits	14	
marker_bits	2	3

図 4 5

【図 4 6】

history stream(40-7)		
	bits	value
num_other_bits	7	
marker_bit	1	1

図 4 6

【図 4 7】

history_stream()	No. of bits	Mnemonic
next_start_code ()		
sequence_header ()		
sequence_extension ()		
extension_and_user_data (0)		
if(nextbits()= group_start_code){		
group_of_pictures_header()		
extension_and_user_data (1)		
}		
picture_header()		
picture_coding_extension()		
re_coding_stream_info()		
extensions_and_user_data (2)		
picture_data()		
sequence_end_code	32	bslbf
}		

図 4 7

【図 48】

sequence_header()	No. of bits	Mnemonic
sequence_header_code()	32	bslbf
horizontal_size_value	12	uimsbf
vertical_size_value	12	uimsbf
aspect_ratio_information	4	uimsbf
frame_rate_code	4	uimsbf
bit_rate_value	18	uimsbf
marker_bit	1	bslbf
vbv_buffer_size_value	10	uimsbf
constrained_parameters_flag	1	bslbf
load_intra_quantiser_matrix	1	uimsbf
if(load_intra_quantiser_matrix)		
intra_quantiser_matrix[64]	8*64	uimsbf
load_non_intra_quantiser_matrix	1	uimsbf
if(load_non_intra_quantiser_matrix)		
non_intra_quantiser_matrix[64]	8*64	uimsbf
next_start_code()		
}		

図 48

【図 49】

sequence_extension()	No. of bits	Mnemonic
extension_start_code	32	bslbf
extension_start_code_identifier	4	uimsbf
profile_and_level_indication	8	uimsbf
progressive_sequence	1	uimsbf
chroma_format	2	uimsbf
horizontal_size_extension	2	uimsbf
vertical_size_extension	2	uimsbf
bit_rate_extension	12	uimsbf
marker_bit	1	bslbf
vbv_buffer_size_extension	8	uimsbf
low_delay	1	uimsbf
frame_rate_extension_n	2	uimsbf
frame_rate_extension_d	5	uimsbf
next_start_code()		
}		

図 49

【図 50】

extension_and_user_data()	No. of bits	Mnemonic
while((nextbits() == extension_start_code) (nextbits() == user_data_start_code)){		
if((i==2)&&(nextbits() == extension_start_code))		
extension_data()		
if(nextbits() == user_data_start_code)		
user_data()		
}		
}		

図 50

【図 51】

user_data()	No. of bits	Mnemonic
user_data_start_code	32	bslbf
while(nextbits() != '0000 0000 0000 0000 0001'){		
user_data	8	uimsbf
}		
next_start_code()		
}		

図 51

【図 52】

group_of_pictures_header()	No. of bits	Mnemonic
group_start_code	32	bslbf
time_code	25	bslbf
closed_gop	1	uimsbf
broken_link	1	uimsbf
next_start_code()		
}		

図 52

【図 53】

picture_header()	No. of bits	Mnemonic
picture_start_code	32	bslbf
temporal_reference	10	uimsbf
picture_coding_type	3	uimsbf
vbv_delay	16	uimsbf
if(picture_coding_type==2 picture_coding_type==3){		
full_pel_forward_vector	1	bslbf
forward_f_code	3	bslbf
}		
if(picture_coding_type==3){		
full_pel_forward_vector	1	bslbf
backward_f_code	3	bslbf
}		
while(nextbits() == '1'){		
extra_bit_picture/*with the value '1'*/	1	uimsbf
extra_information_picture	8	uimsbf
}		
extra_bit_picture/*with the value '0'*/	1	uimsbf
next_start_code()		
}		

図 53

【図 5 4】

picture_coding_extension() {	No. of bits	Mnemonic
extension_start_code	32	bslbf
extension_start_code_identifier	4	uimsbf
f_code [0] [0] /*forward horizontal*/	4	uimsbf
f_code [0] [1] /*forward vertical*/	4	uimsbf
f_code [1] [0] /*backward horizontal*/	4	uimsbf
f_code [1] [1] /*backward vertical*/	4	uimsbf
intra_dc_precision	2	uimsbf
picture_structure	2	uimsbf
top_field_first	1	uimsbf
frame_pred_frame_dct	1	uimsbf
concealment_motion_vectors	1	uimsbf
q_scale_type	1	uimsbf
intra_vlc_format	1	uimsbf
alternate_scan	1	uimsbf
repeat_first_field	1	uimsbf
chroma_420_type	1	uimsbf
progressive_frame	1	uimsbf
composite_display_flag	1	uimsbf
if(composite_display_flag){		
v_axis	1	uimsbf
field_sequence	3	uimsbf
sub_carrier	1	uimsbf
burst_amplitude	7	uimsbf
sub_carrier_phase	8	uimsbf
}		
next_start_code()		
}		

図 5 4

【図 5 5】

extension_data() {	No. of bits	Mnemonic
while(nextbits()==extension_start_code) {		
extension_start_code	32	bslbf
if (nextbits()=="Quant Matrix Extension ID")		
quant_matrix_extension()		
else if (nextbits()=="Copyright Extension ID")		
copyright_extension()		
else		
picture_display_extension()		
}		

図 5 5

【図 5 6】

quant_matrix_extension() {	No. of bits	Mnemonic
extension_start_code_identifier	4	uimsbf
load_intra_quantiser_matrix	1	uimsbf
if (load_intra_quantiser_matrix)		
intra_quantiser_matrix [64]	8*64	uimsbf
load_non_intra_quantiser_matrix	1	uimsbf
if (load_non_intra_quantiser_matrix)		
non_intra_quantiser_matrix [64]	8*64	uimsbf
load_chroma_intra_quantiser_matrix	1	uimsbf
if (load_chroma_intra_quantiser_matrix)		
chroma_intra_quantiser_matrix [64]	8*64	uimsbf
load_chroma_non_intra_quantiser_matrix	1	uimsbf
if (load_chroma_non_intra_quantiser_matrix)		
chroma_non_intra_quantiser_matrix [64]	8*64	uimsbf
next_start_code()		
}		

図 5 6

【図 5 7】

copyright_extension() {	No. of bits	Mnemonic
extension_start_code_identifier	4	uimsbf
copyright_flag	1	bslbf
copyright_identifier	8	uimsbf
original_or_copy	1	bslbf
reserved	7	uimsbf
marker_bit	1	bslbf
copyright_number_1	20	uimsbf
marker_bit	1	bslbf
copyright_number_2	22	uimsbf
marker_bit	1	bslbf
copyright_number_3	22	uimsbf
next_start_code()		
}		

図 5 7

【図 5 8】

picture_display_extension() {	No. of bits	Mnemonic
extension_start_code_identifier	4	uimsbf
for(i=0;i<number_of_frame_centre_offsets;i++){		
frame_centre_horizontal_offset	16	simbsf
marker_bit	1	bslbf
frame_centre_vertical_offset	16	simbsf
marker_bit	1	bslbf
}		
next_start_code()		
}		

図 5 8

【図 5 9】

picture_data() {	No. of bits	Mnemonic
while(nextbits()==slice_start_code) {		
slice()		
}		
next_start_code()		
}		

図 5 9

【図 6 0】

slice() {	No. of bits	Mnemonic
slice_start_code	32	bslbf
slice_quantiser_scale_code	5	uimsbf
if (nextbit()=='1') {		
intra_slice_flag	1	bslbf
intra_slice	1	uimsbf
reserved_bits	7	uimsbf
while(nextbits()=='1') {		
extra_bit_slice/*with the value '1'*/	1	uimsbf
extra_information_slice	8	uimsbf
}		
extra_bit_slice/*with the value '0'*/	1	uimsbf
do {		
macroblock()		
} while(nextbit()!='000 0000 0000 0000 0000 0000')		
next_start_code()		
}		

図 6 0

【図 6 1】

macroblock() {	No. of bits	Mnemonic
while(nextbits()=='0000 0001 000')		
macroblock_escape	11	bslbf
macroblock_address_increment	1-11	vlcibf
macroblock_modes()		
if (macroblock_quant)		
macroblock_quantiser_scale_code	5	uimbsf
if (macroblock_motion_forward		
(macroblock_intra && concealment_motion_vectors))		
motion_vectors(0)		
if (macroblock_motion_backward)		
motion_vectors(1)		
if (macroblock_intra && concealment_motion_vectors)		
marker_bit	1	bslbf
}		

図 6 1

【図 6 2】

macroblock_modes() {	No. of bits	Mnemonic
macroblock_type	1-9	vlcibf
if (macroblock_motion_forward		
macroblock_motion_backward) {		
if (picture_structure=='frame') {		
if (frame_pred_frame_dct==0)		
frame_motion_type	2	uimbsf
} else {		
field_motion_type	2	uimbsf
}		
if ((picture_structure=="Frame picture") &&		
(frame_pred_frame_dct==0) &&		
(dct_type_flag==1)) {		
dct_type	1	uimbsf
}		
}		

図 6 2

【図 6 3】

motion_vectors(s) {	No. of bits	Mnemonic
if (motion_vector_count==1) {		
if ((mv_format==field)&&(dmv!=1)		
motion_vertical_field_select [0][s]	1	uimbsf
motion_vector(0,s)		
} else {		
motion_vertical_field_select [0][s]	1	uimbsf
motion_vector(0,s)		
motion_vertical_field_select [1][s]	1	uimbsf
motion_vector(1,s)		
}		
}		

図 6 3

【図 6 4】

motion_vectors(r,s) {	No. of bits	Mnemonic
motion_code[r][s][0]	1-11	vlcibf
if ((f_code[s][0]!=1)&&(motion_code[r][s][0]!=0)		
motion_residual [r][s][0]	1-8	uimbsf
if (dmv==1)		
dmvector [0]	1-2	vlcibf
motion_code[r][s][1]	1-11	vlcibf
if ((f_code[s][1]!=1)&&(motion_code[r][s][1]!=0)		
motion_residual [r][s][1]	1-8	uimbsf
if (dmv==1)		
dmvector [1]	1-2	vlcibf
}		

図 6 4

【図 6 6】

macroblock_type V L C code					
					macroblock_quant
					dct_type_flag
					macroblock_motion_forward
					macroblock_motion_backward
					Description
1	0	1	1	0	MC, Coded
01	0	1	0	0	No MC, Coded
001	0	0	0	0	MC, Not Coded
0001 1	0	1	0	0	Intra
0001 0	1	1	1	0	MC, Coded, Quant
0000 1	1	1	0	0	No MC, Coded, Quant
0000 01	1	1	0	0	Intra, Quant

図 6 6

【図 6 5】

macroblock_type V L C code					
					macroblock_quant
					dct_type_flag
					macroblock_motion_forward
					macroblock_motion_backward
					Description
1	0	1	0	0	Intra
01	1	1	0	0	Intra, Quant

図 6 5

【図 6 7】

macroblock_type V L C code					
					macroblock_quant
					dct_type_flag
					macroblock_motion_forward
					macroblock_motion_backward
					Description
10	0	0	0	0	Interp, Not Coded
11	0	1	1	1	Interp, Coded
010	0	0	0	0	Bwd, Not Coded
011	0	1	0	1	Bwd, Coded
0010	0	0	0	0	Fwd, Not Coded
0011	0	1	1	0	Fwd, Coded
0001 1	0	1	0	0	Intra
0001 0	1	1	1	1	Interp, Coded, Quant
0000 11	1	1	1	0	Fwd, Coded, Quant
0000 10	1	1	0	1	Bwd, Coded, Quant
0000 01	1	1	0	0	Intra, Quant

図 6 7

【图 7-3】

picture rate elements(72-2)

broken_link	1bit flag	1	205	205	2	
picture header						
temporal_reference	10bit uimsbf	10	206	215	1	
picture_coding_type	3bit uimsbf	3	216	218	1	
vbv_delay	16bit uimsbf	16	219	234	1	should be calculated if not present in stream
full_pel_forward_vector	1bit flag	1	235	235	1	
forward_f_code	3bit uimsbf	3	236	238	1	
full_pel_backward_vector	1bit flag	1	239	239	1	
backward_f_code	3bit uimsbf	3	240	242	1	
picture_coding_extension						
forward_horizontal_f_code	4bit uimsbf	4	243	246	1	
forward_vertical_f_code	4bit uimsbf	4	247	250	1	
backward_horizontal_f_code	4bit uimsbf	4	251	254	1	
backward_vertical_f_code	4bit uimsbf	4	255	258	1	
intra_dc_precision	2bit uimsbf	2	259	260	1	
picture_structure	2bit uimsbf	2	261	262	1	
top_field_first	1bit flag	1	263	263	1	
frame_pred_frame_dct	1bit flag	1	264	264	1	
concealment_motion_vectors	1bit flag	1	265	265	1	
q_scale_type	1bit flag	1	266	266	1	
intra_vlc_format	1bit flag	1	267	267	1	
alternate_scan	1bit flag	1	268	268	1	
repeat_first_field	1bit flag	1	269	269	1	
chroma_420_type	1bit flag	1	270	270	1	
progressive_frame	1bit flag	1	271	271	1	
composite_display_flag	1bit flag	1	272	272	1	
v_axis	1bit flag	1	273	273	1	
field_sequence	3bit uimsbf	3	274	276	1	
sub_carrier	1bit flag	1	277	277	1	
burst_amplitude	7bit uimsbf	7	278	284	1	
sub_carrier_phase	8bit uimsbf	8	285	292	1	
quant_matrix_extension						
load_intra_quantiser_matrix	1bit flag	1	293	293	1	(1)
load_non_intra_quantiser_matrix	1bit flag	1	294	294	1	(1)
load_chroma_intra_quantiser_matrix	1bit flag	1	295	295	1	(1)
load_chroma_non_intra_quantiser_matrix	1bit flag	1	296	296	1	(1)
intra_quantiser_matrix[64]	64*0...255	512	297	808	2	(1)
non_intra_quantiser_matrix[64]	64*0...255	512	809	1320	2	(1)
chroma_intra_quantiser_matrix[64]	64*0...255	512	1321	1832	2	(1)
chroma_non_intra_quantiser_matrix[64]	64*0...255	512	1833	2344	2	(1)
picture_display_extension						
frame_centre_horizontal_offset_1	16bit uimsbf	16	2345	2360	2	
frame_centre_vertical_offset_1	16bit uimsbf	16	2361	2376	2	
frame_centre_horizontal_offset_2	16bit uimsbf	16	2377	2392	2	

图 7-3

【图 7-4】

picture rate elements(72-3)

frame_centre_vertical_offset_2	16bit unsigned	16	2393	2408	2
frame_centre_horizontal_offset_3	16bit unsigned	16	2409	2424	2
frame_centre_vertical_offset_3	16bit unsigned	16	2425	2440	2
copyright_extension					
copyright_flag	1bit flag	1	2441	2441	2
copyright_identifier	8bit code	8	2442	2449	2
original_or_copy	1bit flag	1	2450	2450	2
copyright_number	64bit unsigned	64	2451	2514	2
PTS/DTS					
PTS_DTS_flag	2bit flag	2	2515	2516	1
PTS Value	33bit unsigned	33	2517	2549	2
DTS Value	33bit unsigned	33	2550	2582	2
spare reserved bits					
spare	41bit unsigned	41	2583	2623	
user data area					
user_data		1664	2624	4287	2
picture_rate_information_CRC					
32-bit protection CRT	32bit unsigned	32	4288	4319	

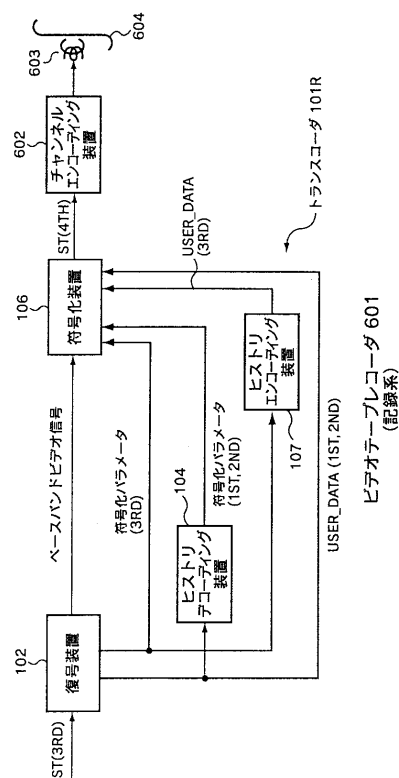
图 7-4

【图 7-5】

D9	Cb[0][9]	Y[0][9]	Cr[0][9]	Y[1][9]	Cb[1][9]	Y[2][9]	Cr[1][9]	Y[3][9]
D8	Cb[0][8]	Y[0][8]	Cr[0][8]	Y[1][8]	Cb[1][8]	Y[2][8]	Cr[1][8]	Y[3][8]
D7	Cb[0][7]	Y[0][7]	Cr[0][7]	Y[1][7]	Cb[1][7]	Y[2][7]	Cr[1][7]	Y[3][7]
D6	Cb[0][6]	Y[0][6]	Cr[0][6]	Y[1][6]	Cb[1][6]	Y[2][6]	Cr[1][6]	Y[3][6]
D5	Cb[0][5]	Y[0][5]	Cr[0][5]	Y[1][5]	Cb[1][5]	Y[2][5]	Cr[1][5]	Y[3][5]
D4	Cb[0][4]	Y[0][4]	Cr[0][4]	Y[1][4]	Cb[1][4]	Y[2][4]	Cr[1][4]	Y[3][4]
D3	Cb[0][3]	Y[0][3]	Cr[0][3]	Y[1][3]	Cb[1][3]	Y[2][3]	Cr[1][3]	Y[3][3]
D2	Cb[0][2]	Y[0][2]	Cr[0][2]	Y[1][2]	Cb[1][2]	Y[2][2]	Cr[1][2]	Y[3][2]
D1	Cb[0][1]	Y[0][1]	Cr[0][1]	Y[1][1]	Cb[1][1]	Y[2][1]	Cr[1][1]	Y[3][1]
D0	Embedded Aligned MPEG-2 Re-Coding Information Rus.	Y[0][0]	Embedded Aligned MPEG-2 Re-Coding Information Rus.	Y[1][0]	Embedded Aligned MPEG-2 Re-Coding Information Rus.	Y[2][0]	Embedded Aligned MPEG-2 Re-Coding Information Rus.	Y[3][0]

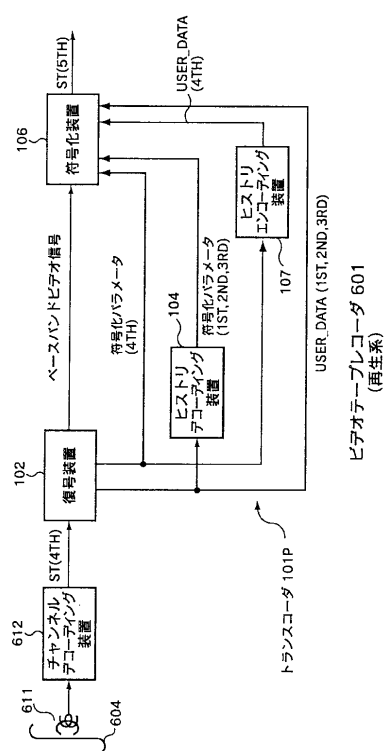
图 7 5

【图 7-6】



976

【图 7-7】

77 ☒

【図 7 8】

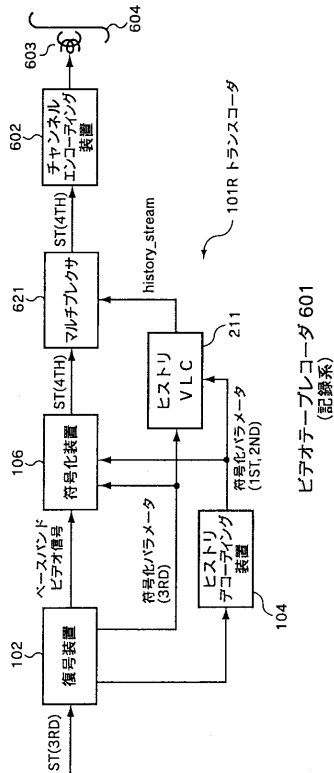
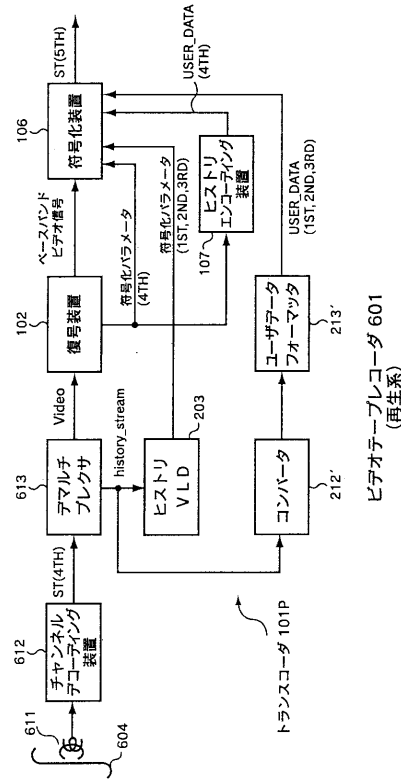


图 78

【图 7 9】



629

【図 80】

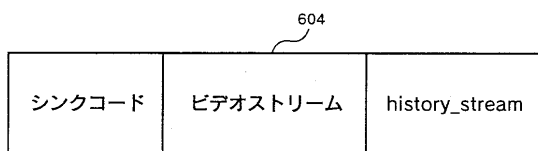


图 80

フロントページの続き

(56)参考文献 欧州特許出願公開第00889650 (EP, A1)

特開平11-074798 (JP, A)

特開平10-145798 (JP, A)

特開平07-288804 (JP, A)

特開平08-065663 (JP, A)

特開平08-111870 (JP, A)

特開平08-130743 (JP, A)

(58)調査した分野(Int.Cl., DB名)

H04N 7/24 — 7/68

H03M 7/30 — 7/50

H04N 5/92