(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号

特許第4543321号
(P4543321)

(45)発行日　平成22年9月15日(2010.9.15)　(24)登録日　平成22年7月9日(2010.7.9)

(51)Int.Cl.			F I		
H04N	***7/26***	***(2006.01)***	H04N	7/13	Z
H04N	***5/92***	***(2006.01)***	H04N	5/92	H

請求項の数 19　(全 88 頁)

(21)出願番号	特願2005-64631(P2005-64631)
(22)出願日	平成17年3月8日(2005.3.8)
(62)分割の表示	特願2000-599213(P2000-599213)の分割
原出願日	平成12年2月9日(2000.2.9)
(65)公開番号	特開2005-253092(P2005-253092A)
(43)公開日	平成17年9月15日(2005.9.15)
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
(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)

(73)特許権者　000002185
ソニー株式会社
東京都港区港南１丁目７番１号
(74)代理人　100082740
弁理士　田辺　恵基
(72)発明者　北村　卓也
東京都品川区北品川６丁目７番３５号ソニー株式会社内

審査官　古川　哲也

最終頁に続く

(54)【発明の名称】再生装置及び方法

(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記録媒体から再生する再生装置において、

前記記録媒体から、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及び前記履歴符号化パラメータの選択的な組み合わせを識別する識別情報を、前記符号化ストリームとともに取得する取得手段と、

前記取得手段により取得された前記識別情報を参照して、前記取得手段により取得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する選択手段と、

前記選択手段により選択された前記再利用符号化パラメータを再利用して、前記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、

を備え、

前記符号化ストリームと前記履歴符号化パラメータとは、前記記録媒体の異なる領域に記録されている

再生装置。

【請求項２】

前記記録媒体は磁気テープであり、

前記符号化ストリームはシンクコードに近い位置に記録されており、前記履歴符号化パラメータは前記シンクコードから前記符号化ストリームの記録位置より離れた位置に記録

されている
請求項１に記載の再生装置。
【請求項３】
前記選択手段は、前記履歴符号化パラメータに含まれるピクチャタイプが変換処理に用いるピクチャタイプと一致した場合のみ、前記履歴符号化パラメータを前記再利用符号化パラメータとして選択する
請求項１に記載の再生装置。
【請求項４】
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、
前記選択手段は、前記履歴符号化パラメータを前記再利用符号化パラメータとして選択しない場合、前記符号化パラメータ算出手段により算出された前記現在の符号化パラメータを、変換処理に利用する前記再利用符号化パラメータとして選択する
請求項３に記載の再生装置。
【請求項５】
前記変換手段により変換された前記再符号化ストリームを出力する出力手段
を更に備える請求項１に記載の再生装置。
【請求項６】
前記識別情報は、前記変換手段の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に記載の再生装置。
【請求項７】
前記識別情報は、前記符号化ストリームが伝送される伝送路、又は、前記符号化ストリームを記録する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に記載の再生装置。
【請求項８】
前記選択手段により選択された前記再利用符号化パラメータ及び前記履歴符号化パラメータのうちのいずれかを、前記変換手段により変換された前記再符号化ストリームとともに出力する出力手段
を更に備える請求項４に記載の再生装置。
【請求項９】
前記変換手段は、前記選択手段により選択された前記再利用符号化パラメータ及び前記履歴符号化パラメータのうちのいずれかを、前記再符号化ストリームに記述する
請求項８に記載の再生装置。
【請求項１０】
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と、
前記生成手段により生成された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、前記変換手段により変換された前記再符号化ストリームとともに出力する出力手段と
を更に備える請求項４に記載の再生装置。
【請求項１１】
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い、前記生成手段により生成された前記符号化履歴情報を、history_stream()で表されるシンタクスとして前記再符号化ストリームに記述する
請求項１０に記載の再生装置。
【請求項１２】
前記識別情報は、前記履歴符号化パラメータの全てを取得するか、前記履歴符号化パラメータのうちの一部を取得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取得する前記履歴符号化パラメータ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に記載の再生装置。
【請求項１３】
前記選択手段は、前記識別情報に含まれる前記第１の識別情報及び前記第２の識別情報を参照して、前記履歴符号化パラメータの中から、前記再利用履歴符号化パラメータを選択する
請求項１２に記載の再生装置。
【請求項１４】
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い、前記第１の識別情報及び前記第２の識別情報を、red_bw_flag及びred_bw_indicatorで表される記述子として前記再符号化ストリームに記述する
請求項１２に記載の再生装置。
【請求項１５】
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に記載の再生装置。
【請求項１６】
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に記載の再生装置。
【請求項１７】
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に記載の再生装置。
【請求項１８】
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１７に記載の再生装置。
【請求項１９】
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記録媒体から再生する再生方法において、
前記記録媒体から、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及び前記履歴符号化パラメータの選択的な組み合わせを識別する識別情報を、前記符号化ストリームとともに取得するステップと、
前記取得ステップにより取得された前記識別情報を参照して、前記取得ステップにより取得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する選択ステップと、
前記選択ステップにより選択された前記再利用符号化パラメータを再利用して、前記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、
を有し、
前記符号化ストリームと前記履歴符号化パラメータとは、前記記録媒体の異なる領域に記録されている
再生方法。
【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ダ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。

【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。

【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。

【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てれいるので、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。

【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。

【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。

【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言う問題があった。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記録媒体から再生する再生装置において、記録媒体から、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及び履歴符号化パラメータの選択的な組み合わせを識別する識別情報を、符号化ストリームとともに取得する取得手段と、取得手段により取得された識別情報を参照して、取得手段により取得された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する選択手段と、選択手段により選択された再利用符号化パラメータを再利用して、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を備え、符号化ストリームと履歴符号化パラメータとは、記録媒体の異なる領域に記録されているようにした。

【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記録媒体から再生する再生方法において、記録媒体から、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及び履歴符号化パラメータの選択的な組み合わせを識別する識別情報を、符号化ストリームとともに取得するステップと、取得ステップにより取得された識別情報を参照して、取得ステップにより取得された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する選択ステップと、選択ステップにより選択された再利用符号化パラメータを再利用して、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を有し、符号化ストリームと履歴符号化パラメータとは、記録媒体の異なる領域に記録されているようにした。

【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。
【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。
【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。
【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】
【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ｔ１乃至ｔ３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ｘ１（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ｘ２（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ｘ１とｘ２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。
【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。
【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。
【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。
【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。

【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。

【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Ａijの総和ΣＡijの絶対値｜ΣＡij｜と、マクロブロックの信号Ａijの絶対値｜Ａij｜の総和Σ｜Ａij｜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Ａijと、予測画像のマクロブロックの信号Ｂijの差Ａij－Ｂijの絶対値｜Ａij－Ｂij｜の総和Σ｜Ａij－Ｂij｜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。

【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】

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。

【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。

【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】

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。

【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６０、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。
【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】
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。
【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】
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ータから、動き補償回路６４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。

【００８３】

スイッチ５３ｄ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）データと、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。

【００８４】

すなわち、動き補償回路６４は、予測判定回路５４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６３ａと後方予測画像部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６４より出力された予測画像データは、演算器５３ｃに供給される。演算器５３ｃは、動きベクトル検出回路５０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６４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。

【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６３には記憶されない。

【００８７】

なお、フレームメモリ６３において、前方予測画像部６３ａと後方予測画像部６３ｂ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８１に一時記憶された後、復号回路９０の可変長復号回路８２に供給される。可変長復号回路８２は、受信バッファ８１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８７に出力し、量子化スケールを逆量子化回路８３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８３に出力する。

【００９０】

逆量子化回路８３は、可変長復号回路８２より供給された画像データを、同じく可変長復号回路８２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８４に出力す

る。逆量子化回路８３より出力されたデータ（DCT係数）は、IDCT回路８４により、逆離散コサイン変換処理が施され、演算器８５に供給される。

【００９１】

IDCT回路８４より演算器８５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８５より出力され、演算器８５に後に入力される画像データ（Ｐ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。

【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）。

【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。

【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。

【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。

【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でけでなく、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ＳＴ（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。

【０１０８】

また別の呼び方として、第３世代の符号化ストリームＳＴ（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ＳＴ（3rd）中に、第３の符号化パラメータを記述するでけでなく、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。

【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。

【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。

【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ＳＴ（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ＳＴ（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。

【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Frame ＤＣＴモードかField ＤＣＴ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quatntiser_scale_codeは、逆量子化回路８３に供給され、picture_coding_type、quatntiser_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、quatntiser_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ビットのマーカービット（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。

【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。

【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。

【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。

【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】

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。

【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。

【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の情報を受け取り、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。

【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。

【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は、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。

【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。

【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。

【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。

【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。

【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った場合には、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った場合には、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】

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いけないものとする。

【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FrameＤＣＴモードかField ＤＣＴモードかを示すdct_type等である。

【０１４１】

コントローラ７０は、ヒストリ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。

【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世代のヒストリ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】

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。

【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。

【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。

【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モード切り替え回路５５がこのdct_typeによって示されるＤＣＴモードに対応した信号処理を行うように、ＤＣＴ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。

【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。

【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。

【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１５[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なない。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。

【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】
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。
【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。
【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。
【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】

また、逆符号語変換器３１４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３１５に出力する。アドレス発生回路３１５は、この情報と逆符号長変換器３１３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、RAM３１１に出力するとともに、シフト量を発生し、バレルシフタ３１２に出力する。

【０１６８】

図２２は、コンバータ２１２の構成例を表している。この例においては、ヒストリVLC２１１とコンバータ２１２の間に配置されているバッファメモリ３２０の、コントローラ３２６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－ＦＦ）３２１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３２１より読み出されたデータは、スタッフ回路３２３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３２２にも供給され、保持される。Ｄ型フリップフロップ３２２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３２１より読み出された８ビットのデータと合成され、１６ビットのパラレルデータとして、スタッフ回路３２３に供給される。

【０１６９】

スタッフ回路３２３は、コントローラ３２６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（スタッフィングし）、合計１７ビットのデータとして、バレルシフタ３２４に出力する。

【０１７０】

バレルシフタ３２４は、コントローラ３２６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３２５に出力する。Ｄ型フリップフロップ３２５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３２７を介して、後段のユーザデータフォーマッタ２１３に供給される。この時、コントローラ３２６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２１２とユーザデータフォーマッタ２１３との間に介在するバッファメモリ３２７に供給する。

【０１７１】

図２３は、スタッフ回路３２３の構成例を表している。Ｄ型フリップフロップ３２２，３２１より入力された１６ビットのデータは、それぞれスイッチ３３１－１６乃至３３１－１の接点ａに入力されている。スイッチ３３１－ｉ（ｉ＝０乃至１５）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３３１－１２の接点ｃには、MSB側に隣接するスイッチ３３１－１３の接点ａに供給されているLSBから１３番目のデータが供給されており、スイッチ３３１－１３の接点ｃには、MSB側に隣接するスイッチ３３１－１４の接点ａに供給されているLSB側から１４番目のデータが供給されている。

【０１７２】

但し、LSBに対応するスイッチ３３１－１よりさらに下側のスイッチ３３１－０の接点ａは、開放されている。また、MSBに対応するスイッチ３３１－１６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。

【０１７３】

各スイッチ３３１－０乃至３３１－１６の接点ｂには、データ”１”が供給されている。

【０１７４】

デコーダ３３２は、コントローラ３２６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３３１－０乃至３３１－１６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。

【０１７５】

図２３は、LSB側から１３番目にデータ”１”を挿入する場合の例を示している。従って、この場合、スイッチ３３１－０乃至スイッチ３３１－１２は、いずれも接点ｃ側に切り替えられ、スイッチ３３１－１３は、接点ｂ側に切り替えられ、スイッチ３３１－１４乃至スイッチ３３１－１６は、接点ａ側に切り替えられている。

【０１７６】

図２２のコンバータ２１２は、以上のような構成により、２２ビットの符号を２３ビットに変換して、出力することになる。

【０１７７】

図２４は、図２２のコンバータ２１２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２１２のコントローラ３２６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２４（Ａ））を発生すると、バッファメモリ３２０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３２１に一旦保持される。そして、Ｄ型フリップフロップ３２１より読み出されたデータ（図２４（Ｂ））は、スタッフ回路３２３に供給されるとともに、Ｄ型フリップフロップ３２２に供給され、保持される。Ｄ型フリップフロップ３２２に保持されたデータは、そこからさらに読み出され（図２４（Ｃ））、スタッフ回路３２３に供給される。

【０１７８】

従って、スタッフ回路３２３の入力（図２４（Ｄ））は、読み出しアドレスＡ１のタイミングにおいて、最初の１バイトのデータＤ０とされ、次の読み出しアドレスＡ２のタイミングにおいて、１バイトのデータＤ０と１バイトのデータＤ１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Ａ３のタイミングにおいては、データＤ１とデータＤ２より構成される２バイトのデータとなる。

【０１７９】

スタッフ回路３２３には、データ”１”を挿入する位置を示す信号stuff position（図２４（Ｅ））がコントローラ３２６より供給される。スタッフ回路３２３のデコーダ３３２は、スイッチ３３１－１６乃至３３１－０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３２３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２４（Ｆ））が出力される。

【０１８０】

バレルシフタ３２４は、入力されたデータを、コントローラ３２６より供給される信号shift（図２４（Ｇ））で示される量だけバレルシフトして、出力する（図２４（Ｈ））。この出力がさらにＤ型フリップフロップ３２５で一旦保持された後、後段に出力される（図２４（Ｉ））。

【０１８１】

Ｄ型フリップフロップ３２５より出力されるデータには、２２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２２となる。

【０１８２】

図２５は、コンバータ２０２の構成例を表している。このコンバータ２０２のＤ型フリップフロップ３４１乃至コントローラ３４６よりなる構成は、図２２に示したコンバータ２１２のＤ型フリップフロップ３２１乃至コントローラ３２６と基本的に同様の構成であるが、コンバータ２１２におけるスタッフ回路３２３に代えて、ディリート回路３４３が挿入されている点がコンバータ２１２における場合と異なっている。その他の構成は、図２２のコンバータ２１２における場合と同様である。

【０１８３】
すなわち、このコンバータ２０２においては、コントローラ３４６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、ディリート回路３４３が、そのビット（図２２のスタッフ回路３２３で挿入されたデータ”１”）が削除される。
【０１８４】
その他の動作は、図２２のコンバータ２１２における場合と同様である。
【０１８５】
図２６は、ディリート回路３４３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３４２，３４１より入力された１６ビットのデータのうち、LSB側の１５ビットが、それぞれ対応するスイッチ３５１－０乃至３５１－１４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３５２は、コントローラ３４６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、１５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２６は、LSBから第１３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３５１－０乃至スイッチ３５１－１１が接点ａ側に切り替えられ、LSBから第１２番目までの１２ビットが、そのまま選択、出力されている。また、スイッチ３５１－１２乃至３５１－１４は、それぞれ接点ｂ側に切り替えられているので、第１４番目乃至第１６番目のデータが、第１３番目乃至第１５番目のビットのデータとして選択、出力される。
【０１８７】
図２３のスタッフ回路３２３および図２６のディリート回路３４３の入力が１６ビットとなっているのは、それぞれ図２２のコンバータ２１２のスタッフ回路３２３の入力が、Ｄ型フリップフロップ３２２，３２１より供給される１６ビットとされており、また、図２５のコンバータ２０２においても、ディリート回路３４３の入力が、Ｄ型フリップフロップ３４２，３４１により１６ビットとされているためである。図２２において、スタッフ回路３２３の出力する１７ビットをバレルシフタ３２４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２５のコンバータ２０２においても、ディリート回路３４３の出力する１５ビットのデータを、バレルシフタ３４４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２７は、コンバータ２１２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３６１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３２６に出力するようになされている。コントローラ３２６は、例えばカウンタ３６１が連続する０のビットを２２個カウントしたとき、信号stuff positionをスタッフ回路３２３に出力する。また、このとき、コントローラ３２６は、カウンタ３６１をリセットし、再び連続する０のビットの数をカウンタ３６１にカウントさせる。
【０１８９】
その他の構成と動作は、図２２における場合と同様である。
【０１９０】
図２８は、コンバータ２０２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３７１がカウントし、そのカウント結果をコントローラ３４６に出力するようになされている。カウンタ３７１のカウント値が２２に達したとき、コントローラ３４６は、信号delete positionをディリート回路３４３に出力するとともに、カウンタ３７１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３７１にカウントさせる。その他の構成は、図２５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２７と図２８に示す構成は、図２２と図２５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２９は、ユーザデータフォーマッタ２１３の構成例を表している。この例においては、コントローラ３８３がコンバータ２１２とユーザデータフォーマッタ２１３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２１３のスイッチ３８２の接点ａ側に供給される。ROM３８１には、ユーザデータスタートコード、データＩＤ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３１３は、所定のタイミングにおいて、スイッチ３８２を接点ａ側または接点ｂ側に切り替え、ROM３８１に記憶されているデータ、またはコンバータ２１２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１０６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２０１は、図２９のROM３８１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。
【０１９５】
図３０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１０１－１乃至１０１－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１０１－ｉ（ｉ＝１乃至Ｎ）のヒストリ情報多重化装置１０３－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１８）。
【０１９６】
各符号化装置１０６－ｉの符号化装置１０６－ｉ（図１９）は、その可変長符号化回路５８において、ヒストリ情報分離装置１０５－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５７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５８はまた、ヒストリエンコーディング装置１０７－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１８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１０６－ｉの出力するビットストリームは、SDTI(Serial Data Transfer Interface)１０８－ｉを介して、後段のトランスコーダ１０１－（ｉ＋１）に入力される。
【０１９８】
トランスコーダ１０１－ｉとトランスコーダ１０１－（ｉ＋１）は、それぞれ図１５に示すように構成されている。従って、その処理は、図１５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。
【０２００】
図１８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３１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１６に示した復号化装置１０２、および図１９に示した符号化装置１０６は、現在（最新）、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）ヒストリ情報分離装置１０５に出力する。ヒストリ情報分離装置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。

【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、ヒストリ情報分離装置１０５により実行される。

【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）がヒストリ情報分離装置１０５に入力される。

【０２０５】

ステップＳ２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、ヒストリ情報分離装置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。

【０２０７】

ステップＳ４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、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、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Ｂ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（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。

【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、ヒストリ情報分離装置１０５は、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Ｓ７に進む。
【０２１２】
ステップＳ７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Ｓ６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Ｓ８に進む。ステップＳ８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Ｓ４，Ｓ５，Ｓ７，Ｓ８の処理の次にステップＳ９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が変更される。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４MbpsのLong GOP（第１世代）から、N=1，M=1のフレームで構成される５０Mbps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４Mbps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４MbpsのLong GOP（第１世代）から、N=2，M=2である１８Mbps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５０MbpsのShort GOP（第３世代）に変換されて、１Mbps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、４Mbps，１８Mbps、または５０Mbps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。
【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。
【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。
【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。
【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、ヒストリ情報中に存在しないと判定された場合、ステップＳ１８に進む。

【０２３２】

ステップＳ１８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。

【０２３３】

ステップＳ１９において、量子化回路５７は、Feedback_q_scale_codeを用いて量子化を実行する。

【０２３４】

ステップＳ２０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Ｓ１８に戻り、ステップＳ１８乃至Ｓ２０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。

【０２３５】

なお、本実施の形態におけるトランスコーダ１０１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３７に示すように、復号装置１０２と符号化装置１０６を直接接続する（密結合する）ようにしてもよい。

【０２３６】

図１５において説明したトランスコーダ１０１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１０６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３７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バス）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。

【０２３７】

つまり、図３７に示した、復号装置１０２、ヒストリデコーディング装置１０４、符号化装置１０６及びヒストリエンコーディング装置１０７は、図１５において説明した復号装置１０２、ヒストリデコーディング装置１０４、符号化装置１０６及びヒストリエンコーディング装置１０７とまったく同じ機能及び構成を有している。

【０２３８】

復号装置１０２の可変長復号回路１１２は、第３世代の符号化ストリーム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、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１０７及び符号化装置１０６のコントローラ７０にそれぞれ供給する。ヒストリエンコーディング装置１０７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１０６の可変長符号化回路５８に供給する。

【０２３９】

さらに可変長復号回路１１２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１０４及び符号化装置１０６の可変長符号化回路５８に供給する。ヒストリデコーディング装置１０４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１０６のコントローラに供給する。

【０２４０】

符号化装置１０６のコントローラ７０は、ヒストリデコーディング装置１０４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１０２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１０６の符号化処理をコントロールする。

【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。

【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。

【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。

【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。

【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】

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。

【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。

【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。

【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。

【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。

【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エ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。

【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。

【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】

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エレメントは、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エレメントが、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）を表わすデータエレメントは、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。

【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_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。

【０２７８】

horizontal_size_extensionは、シーケンスヘッダのhorizntal_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。

【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_coeffients、display_horizontal_size、及びdisplay_vertical_sizeから構成される。

【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_coeffients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。

【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を示すマクロブロックアサイメントデータ（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を示す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。

【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を示すデータである。

【０２８６】

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。

【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。

【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。

【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_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。

【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。

【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_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。

【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。

【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。

【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】

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】

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_itentifier、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示す開始コードである。extension_start_code_itentifier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。

【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。

【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。

【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。

【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_indicato及びデータセットに関しては、詳しくは後述する。

【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、mocroblock_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。

【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な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る。

【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。mocroblock_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、時間スケー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なう）を維持しなければいけない。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の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_vertical_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。

【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ンタックスについて、図４７乃至図６７を参照して説明する。

【０３２８】

この可変長の履歴ストリームは、next_start_code()関数、sequence_header()関数、sequence_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_extension()関数、re_coding_stream_info()関数、extension_and_user_data(2)関数、及びpicture_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。

【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_code、sequence_header_present_flag、horizontal_size_value、vertical_size_value、aspect_ratio_information、frame_rate_code、bit_rate_value、marker_bit、VBV_buffer_size_value、constrained_parameter_flag、load_intra_quantiser_matrix、intra_quantiser_matrix、load_non_intra_quantiser_matrix、及びnon_intra_quantiser_matrix等である。

【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。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。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】

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。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かを示すスデータである。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horizntal_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５２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。

【０３４０】

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_linkから構成される。

【０３４１】

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。

【０３４２】

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。

【０３４３】

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。

【０３４４】

picture_headr()関数によって定義されたデータエレメントは、図５３に示すように、picture_start_code、temporal_reference、picture_coding_type、vbv_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。

【０３４５】

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。 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。

【０３４６】

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５４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エトとを、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。

【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。

【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。

【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_itentifier、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。

【０３５８】
extension_start_codeは、コピーライトエクステンションのスタート示す開始コードである。extension_start_code_itentifier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_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。

【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。

【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。

【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。

【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。

【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、macroblock()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、マクログ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」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。
【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番めのベクトル又は第２番め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～＋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、ディユ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番めのベクトル又は第２番め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。

【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。

【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。

【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。

【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することができる。

【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】
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。具体的には、図６９に示すように、red_bw_flagが１である場合、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。

【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。

【０４０４】

picture単位の情報は、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。

【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単位の情報は常に伝送するが、macroblock単位の情報は、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。

【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、macroblock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flame_dctかを示すフラグである。

【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。

【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。

【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。

【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[][], mv[][][]），（mb_mfwd, mb_mbwd），（mb_pattern），（coded_block_patt

ern），（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】

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】

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。

【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。

【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。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。

【０４２６】

macroblock_modes()関数のシンタックス要素（図６２参照）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。

【０４２７】

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。

【０４２８】

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。

【０４２９】

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。

【０４３０】

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。

【０４３１】

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。

【０４３２】

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１０１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１０１の処理について、図１５に示したトランスコーダ１０１を参照して説明する。

【０４３３】

この図１５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ＳＴ（３ｒｄ）を受け取り、この符号化ストリームＳＴ（３ｒｄ）のＧＯＰ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ＳＴ（４ｔｈ）を生成するための例である。

【０４３４】

まず最初に、復号装置１０２は、第３世代の符号化ストリームＳＴ（３ｒｄ）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（３th）を、第３世代の符号化ストリームＳＴ（３ｒｄ）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ＳＴ（３ｒｄ）ををデコードし、ベースバンドビデオ信号を生成する。さらに、復号装置１０２は、抽出した第３世代のコーディングパラメータ（３th）を、ヒストリ情報多重化装置１０３に出力する。さらに、復号装置１０２は、第３世代の符号化ストリームＳＴ（３ｒｄ）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１０４に供給する。

【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ジング（parsing）。尚、パージングとは、ストリームの構文を解釈することである。

【０４３６】

このパージング処理を行う際、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、このヒストリー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に供給する。

【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。

【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つのデータセットの中から選択する。

【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】

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_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、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 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、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 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、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、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histry_stream()として伝送するのでは無く、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、histry_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histry_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histry_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histry_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】

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は、色差信号のフィルタリング処理が実行されていないことを表す。

【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。

【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。

【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。

【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。

【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。

【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、histroy_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。

【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’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converted_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。

【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。

【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。

【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。
【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。
【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。
【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。
【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になどに利用できる。
【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。
【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１８の構成を示すブロック図である。
【図８】図８は、図７の予測モード切換回路５２の動作を説明する図である。
【図９】図９は、図７の予測モード切換回路５２の動作を説明する図である。
【図１０】図１０は、図７の予測モード切換回路５２の動作を説明する図である。
【図１１】図１１は、図７の予測モード切換回路５２の動作を説明する図である。
【図１２】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。
【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。
【図１４】図１４は、本発明を適用したトランスコーダ１０１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１０１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１０２に内蔵される復号化装置１０２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１０６に内蔵される符号化装置１０６の構成を示すブロック図である。
【図２０】図２０は、図１５のヒストリVLC２１１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VLD２０３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２１２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３２３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２１２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２０２の構成例を示すブロック図である。
【図２６】図２６は、図２５のディリート回路３４３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２１２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２０２の他の構成例を示すブロック図である。
【図２９】図２９は、図１５のユーザデータフォーマッタ２１３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１０１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１０６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１０６の量子化制御処理を説明する図である。
【図３６】図３６は、図１４の符号化装置１０６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１０１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明する図である。
【図４２】図４２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４３】図４３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４４】図４４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４５】図４５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４６】図４６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４７】図４７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４８】図４８は、sequence_header()のシンタックスを説明する図である。
【図４９】図４９は、sequence_extension()のシンタックスを説明する図である。
【図５０】図５０は、extension_and_user_data()のシンタックスを説明する図である。
【図５１】図５１は、user_data()のシンタックスを説明する図である。
【図５２】図５２は、group_of_pictures_header()のシンタックスを説明する図である。
【図５３】図５３は、picture_header()のシンタックスを説明する図である。
【図５４】図５４は、picture_coding_extension()のシンタックスを説明する図である。
【図５５】図５５は、extension_data()のシンタックスを説明する図である。
【図５６】図５６は、quant_matrix_extension()のシンタックスを説明する図である。
【図５７】図５７は、copyright_extension()のシンタックスを説明する図である。
【図５８】図５８は、picture_display_extension()のシンタックスを説明する図である。
。
【図５９】図５９は、picture_data()のシンタックスを説明する図である。
【図６０】図６０は、slice()のシンタックスを説明する図である。
【図６１】図６１は、macroblock()のシンタックスを説明する図である。
【図６２】図６２は、macroblock_modes()のシンタックスを説明する図である。
【図６３】図６３は、motion_vectors(s)のシンタックスを説明する図である。
【図６４】図６４は、motion_vector(r,s)のシンタックスを説明する図である。
【図６５】図６５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６６】図６６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６７】図６７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６８】図６８は、re_coding_stream_info()のシンタックスを説明する図である。
【図６９】図６９は、red_bw_flag, red_bw_indicatorを説明する図である。
【図７０】図７０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。
【図７１】図７１は、Re_Coding Information Bus macroblock formationを説明する図である。
【図７２】図７２は、Picture rate elementsを説明する図である。
【図７３】図７３は、Picture rate elementsを説明する図である。

【図７４】図７４は、Picture rate elementsを説明する図である。
【図７５】図７５は、Re_Coding Information Busが記録される領域を説明する図である。
【図７６】図７６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７７】図７７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。
【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……フレームメモリー、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。

【図１】

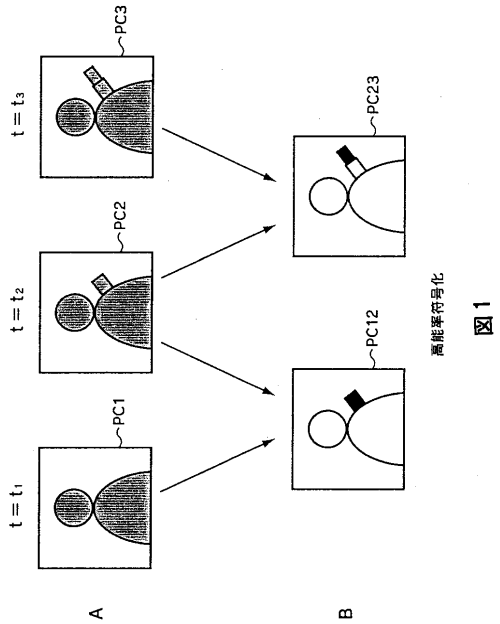


図１

【図２】

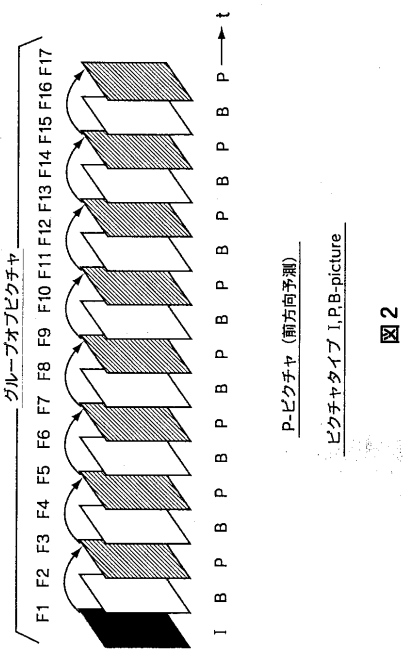


図２

【図３】

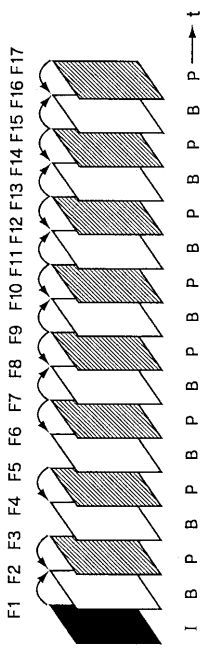


図３

【図４】

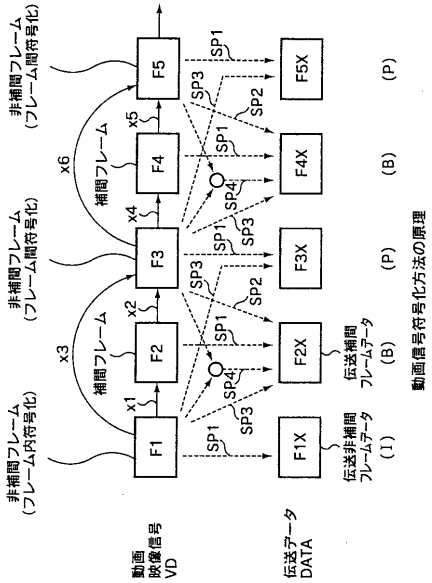
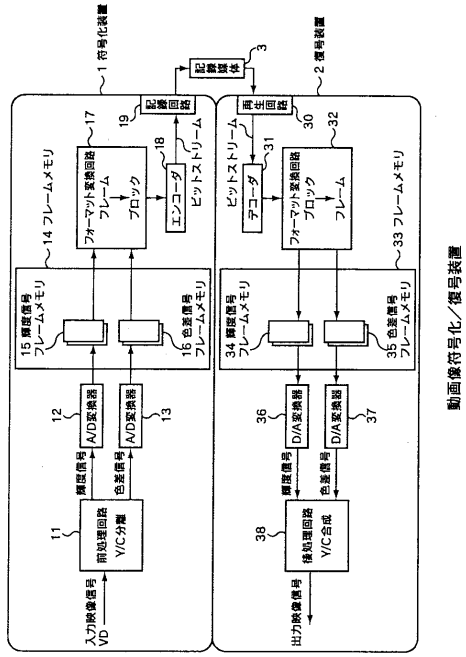


図４

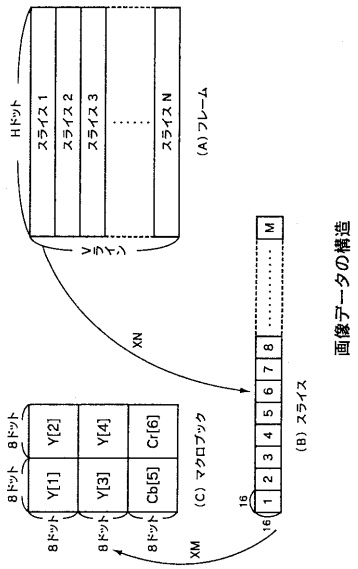
【図５】



動画像符号化／復号装置

図５

【図６】



画像データの構造

図６

【図７】

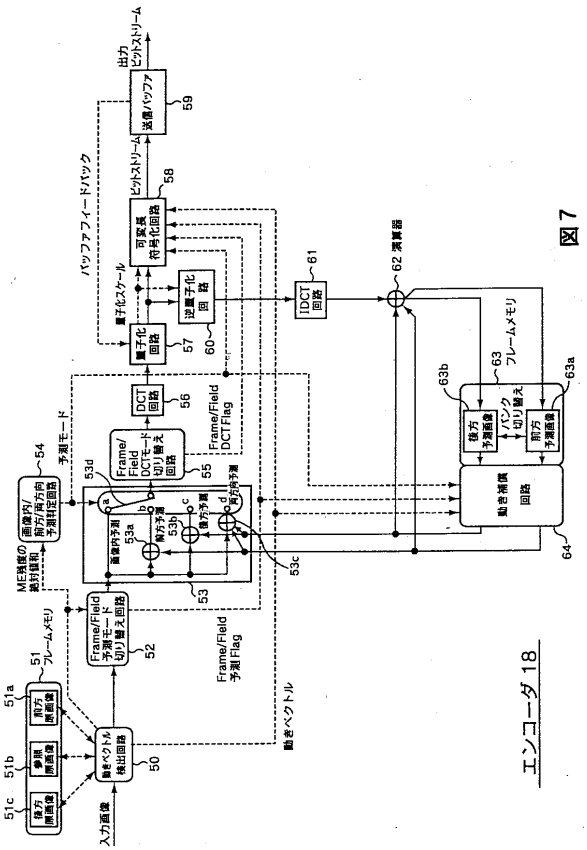


図７

エンコーダ 18

【図８】

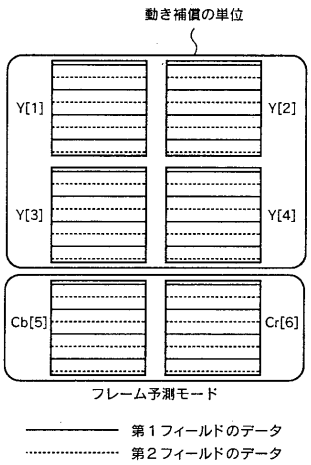


図８

【図９】

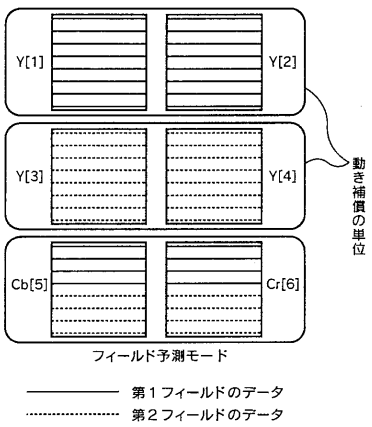


図９

【図１０】

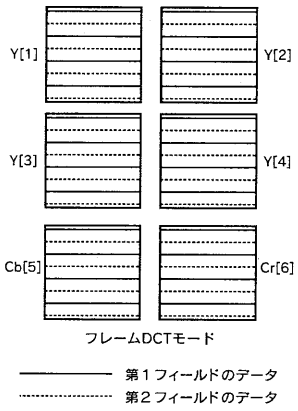


図１０

【図１１】

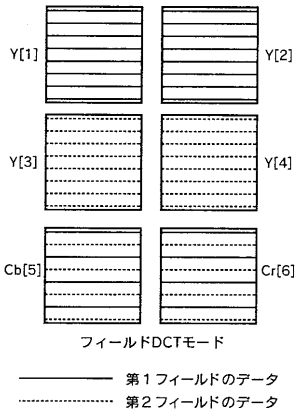


図１１

【図１２】

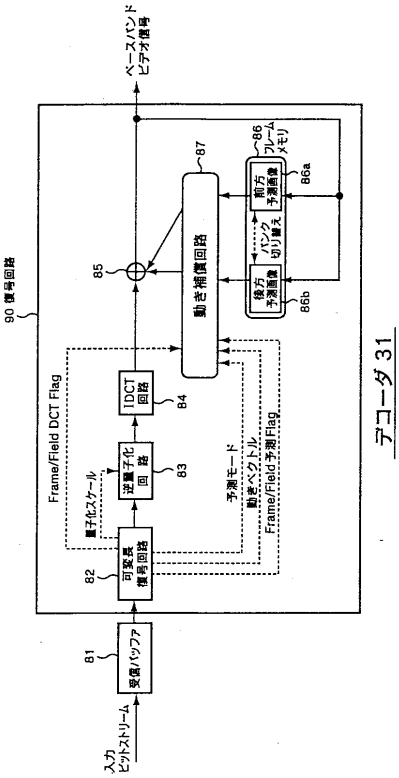


図１２

【図１３】

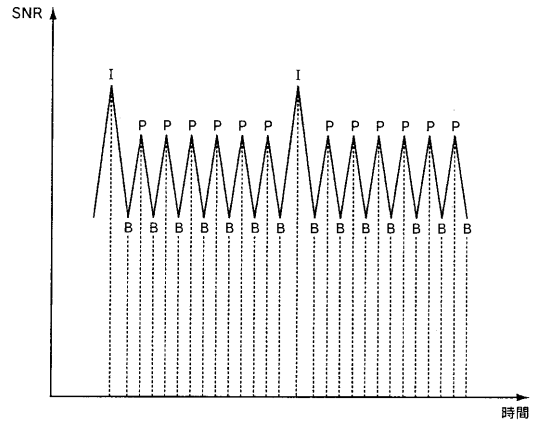


図１３

【図１４】

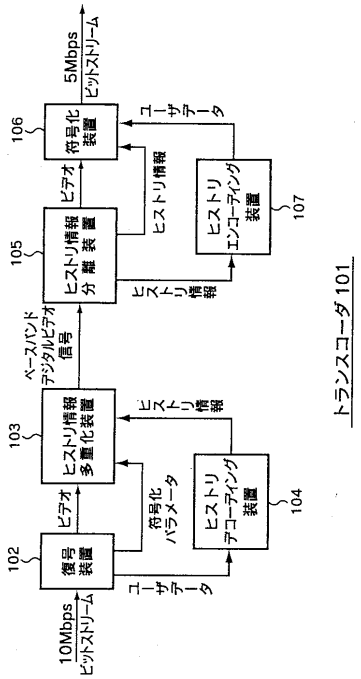


図１４

【図１５】

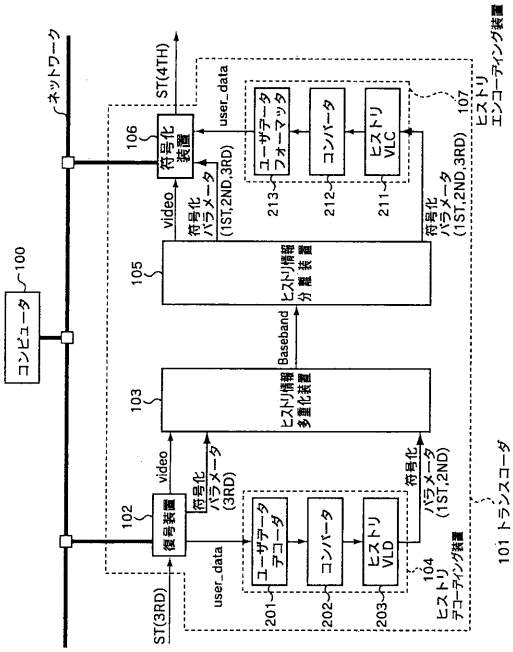


図１５

【図１６】

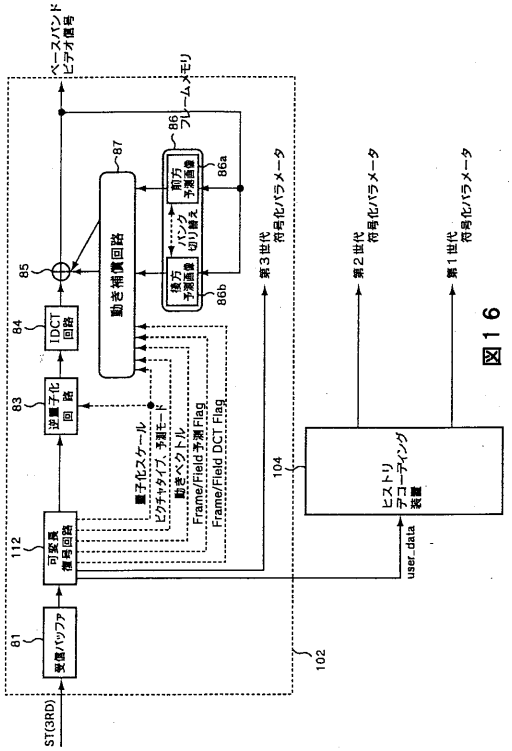


図１６

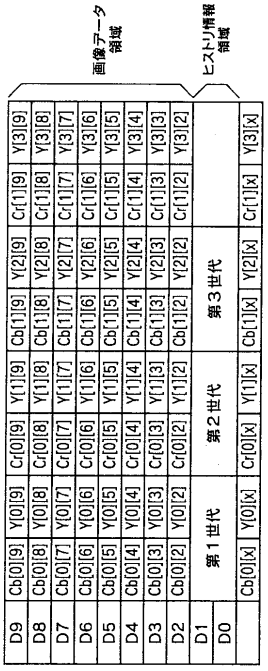
【図１７】

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】



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	第1世代		第2世代		第3世代														
D0	Cb[0][x]	Y[0][x]	Cr[0][x]	Y[1][x]	Cb[1][x]	Y[2][x]	Cr[1][x]	Y[3][x]											

図１８

【図１９】

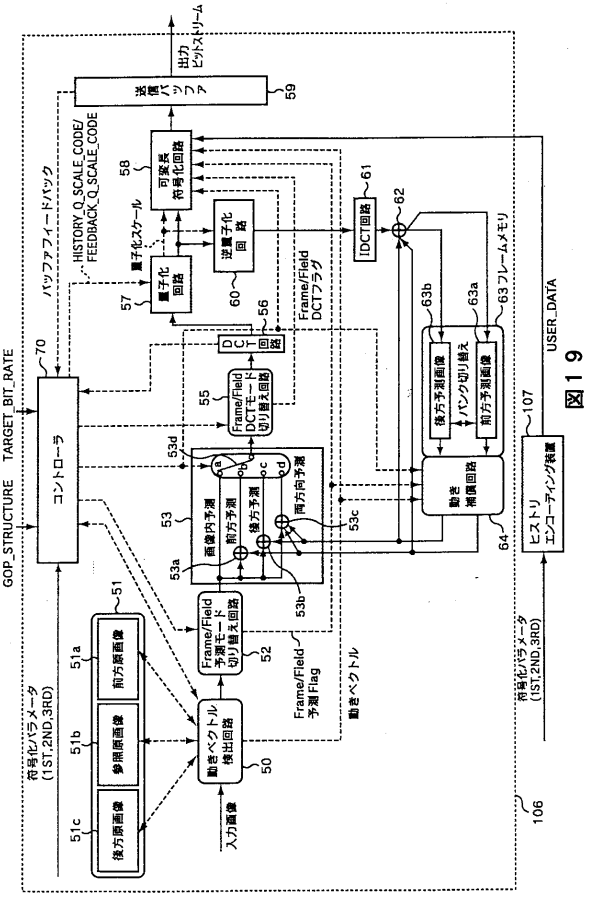
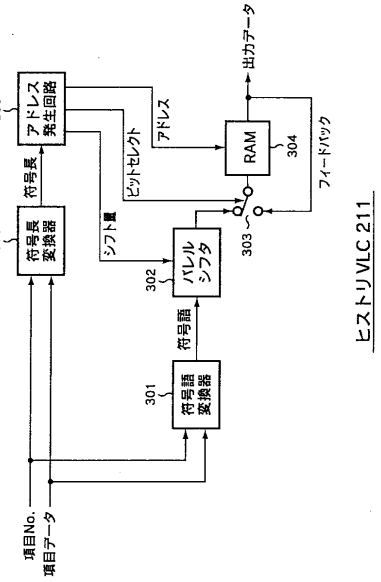


図１９

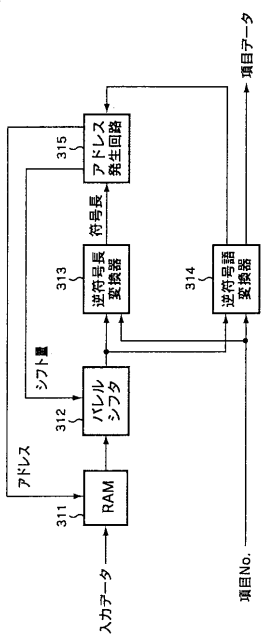
【図２０】



ヒストリVLC 211

図２０

【図２１】



ヒストリVLD 203

図２１

【図２２】

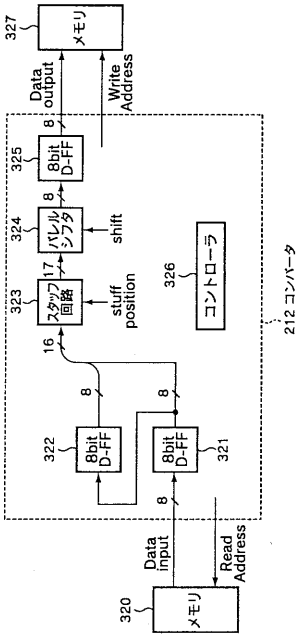


図２２

【図２３】

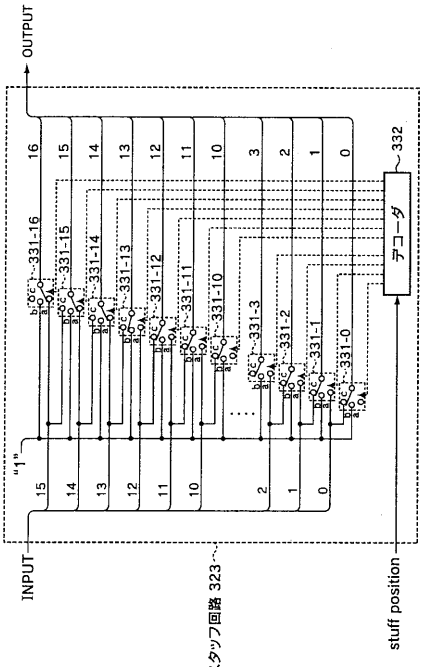


図２３

【図２４】

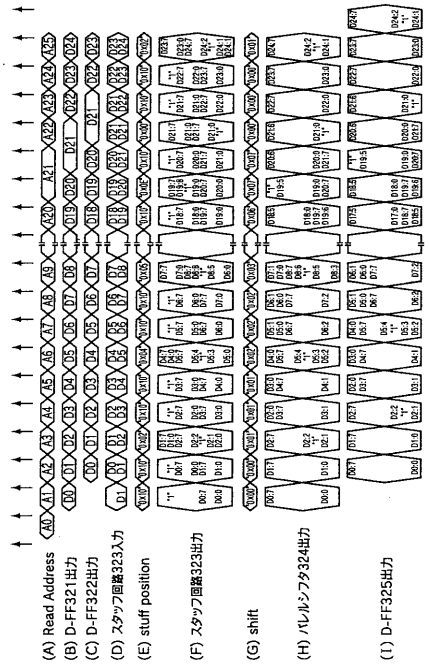


図２４

【図２５】

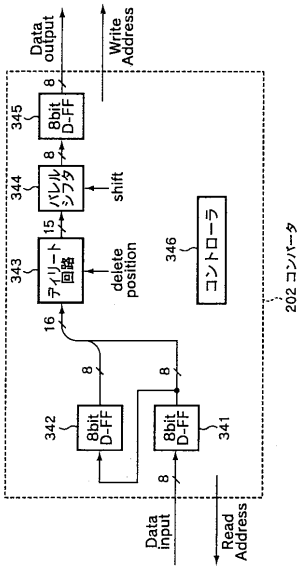


図２５

【図２６】

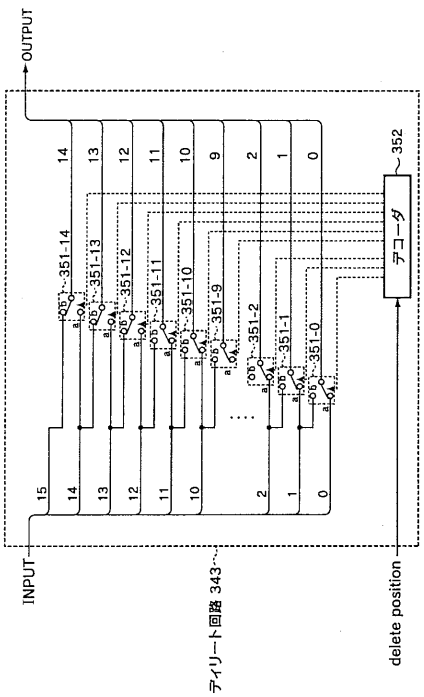


図２６

【図２７】

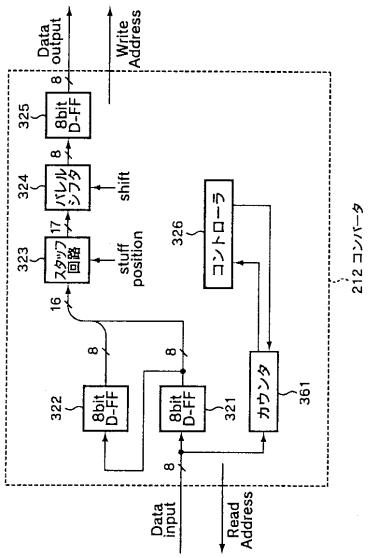


図２７

【図２８】

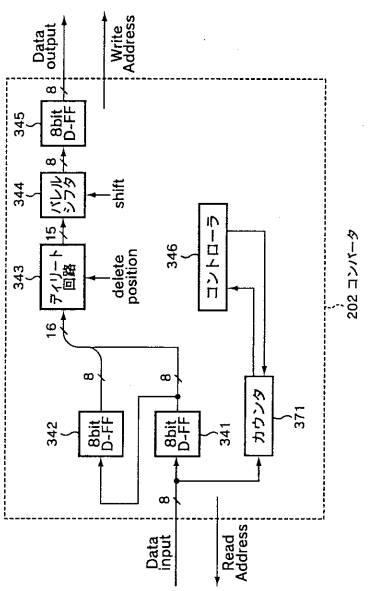


図２８

【図２９】

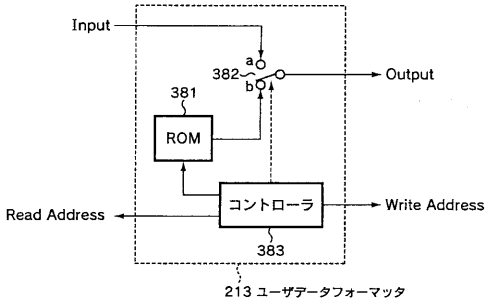


図２９

【図３０】

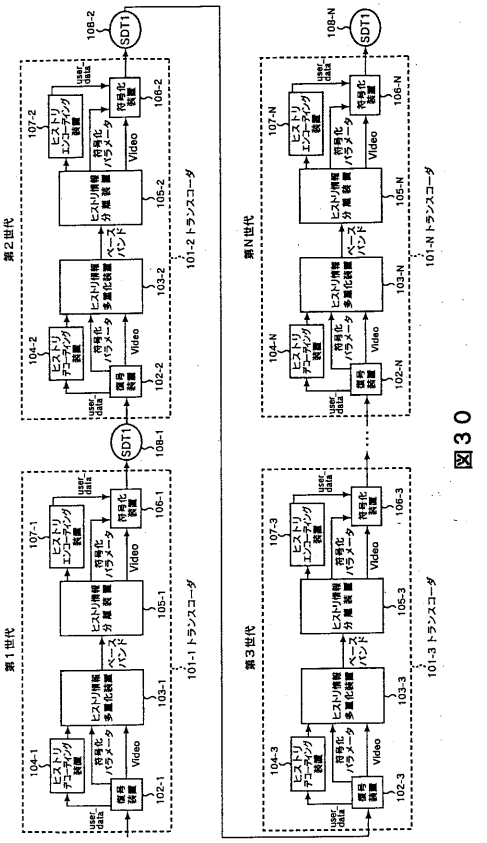


図３０

【図３１】

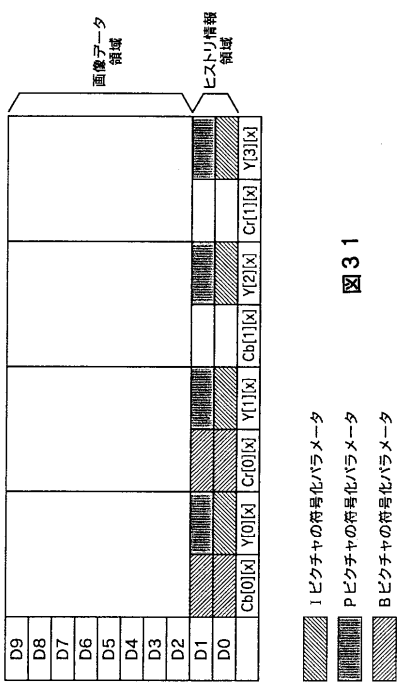


図３１

【図３２】

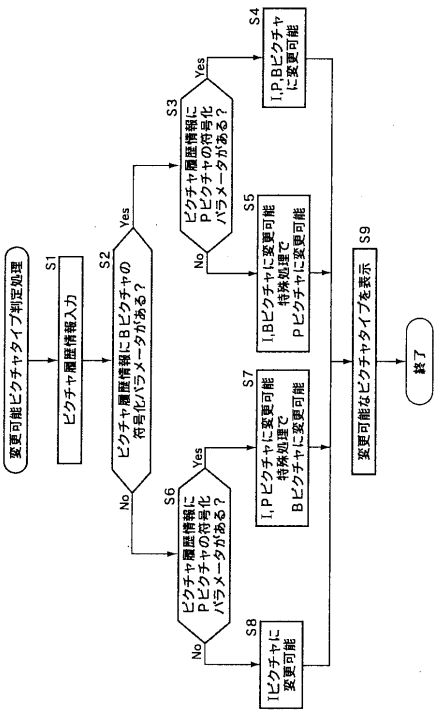


図３２

【図３３】

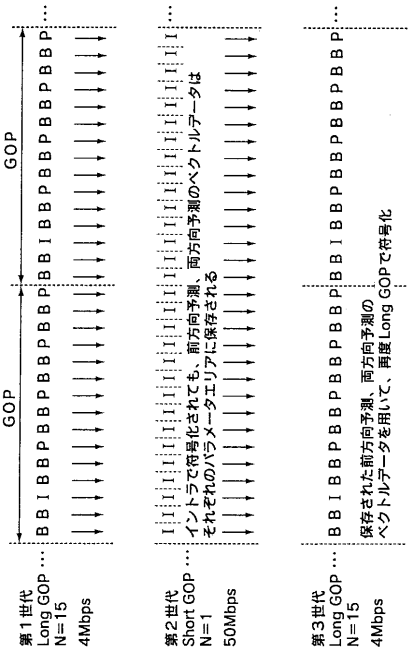


図３３

【図３４】

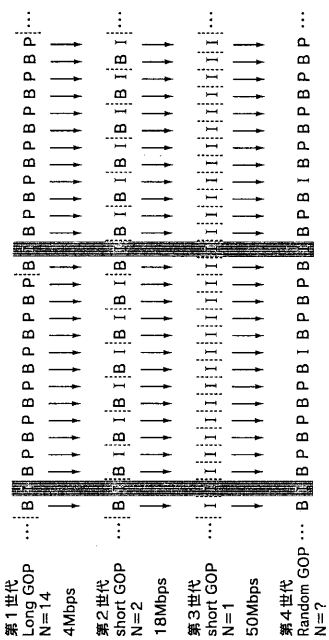


図３４

【図３５】

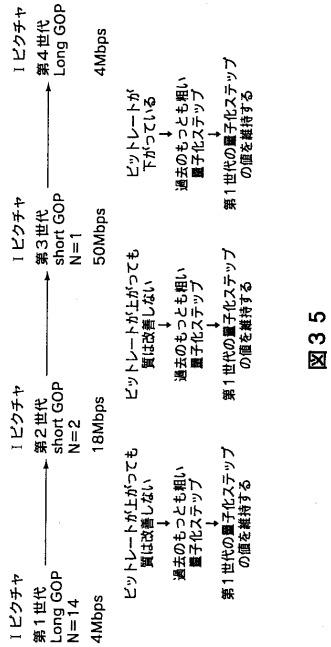


図３５

【図３６】

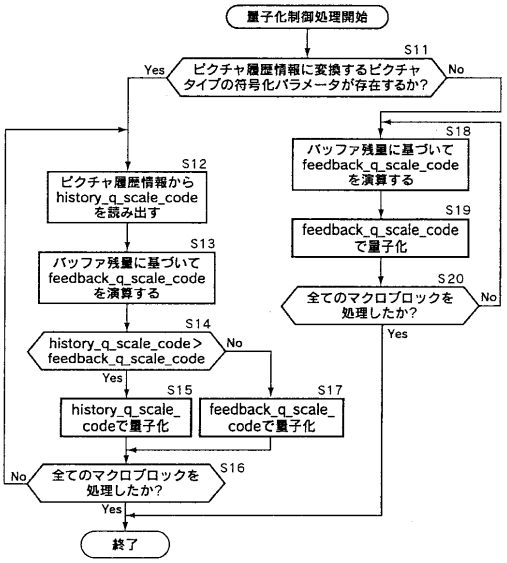


図３６

【図３７】

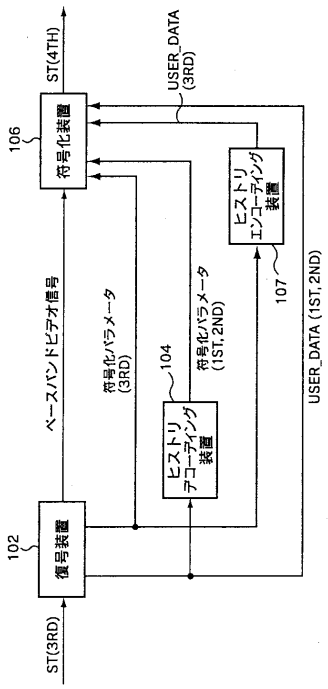


図３７

【図３８】

stream with history data

video_sequence ( ){	No. of bits	Mnemonic
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( )= =groupe_start_code))		
if(nextbits( ) !=sequence_end_code){		
sequence_header ( )		
sequence_extension ( )		
}		
}while(nextbits( ) !=sequence_end_code)		
sequence_end_code		
}	32	bslbf

図３８

【図３９】

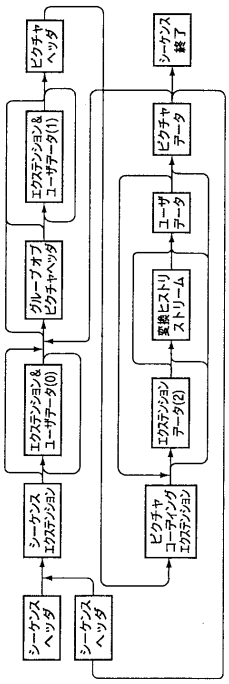


図３９

【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４１】

history stream(40-2)

	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

図４１

【図４２】

history stream(40-3)

	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	
repeart_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	

図４２

【図４３】

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４３

【図４４】

history stream(40-5)

	bits	value
picture_display_extension		
extension_start_code	32	00000185
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４４

【図４５】

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４５

【図４６】

history stream(40-7)

	bits	value
num_other_bits	7	
marker_bit	1	1
}		

図４６

【図４７】

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４７

【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 (i){	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 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５４】

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５４

【図５５】

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５５

【図５６】

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５６

【図５７】

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５７

【図５８】

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	simsbf
marker_bit	1	bslbf
frame_centre_vertical_offset	16	simsbf
marker_bit	1	bslbf
}		
next_start_code( )		
}		

図５８

【図５９】

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

図５９

【図６０】

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６０

【図６１】

macroblock( ) {	No. of bits	Mnemonic
while(nextbits( )=='0000 0001 000')		
macroblock_escape	11	bslbf
macroblock_address_increment	1-11	vlclbf
macroblock_modes( )		
if (macroblock_quant)		
macroblock_quantiser_scale_code	5	uimsbf
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６１

【図６２】

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

図６２

【図６３】

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	uimsbf
motion_vector(0,s)		
} else {		
motion_vertical_field_select [0][s]	1	uimsbf
motion_vector(0,s)		
motion_vertical_field_select [1][s]	1	uimsbf
motion_vector(1,s)		
}		
}		

図６３

【図６４】

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

図６４

【図６５】

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６５

【図６６】

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６６

【図６７】

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６７

【図６８】

re_coding_stream_info ( ) {	No. of bits	Mnemonic
user_data_start_code	32	bslbf
re_coding_stream_info_ID	16	bslbf
red_bw_flag	1	uimsbf
if (red_bw_flag)		
red_bw_indicator	2	uimsbf
if (!red_bw_flag) {		
for(i=0;i<number_of_macroblock;++) {		
marker_bit	3	bslbf
num_other_bits	7	uimsbf
num_mv_bits	8	uimsbf
num_coef_bits	14	uimsbf
}		
}		
next_start_code( )		
}		

図６８

【図６９】

Data Set	red_bw_flag	red_bw_indicator
Data Set1	0	–
Data Set2	1	0
Data Set3	1	1
Data Set4	1	2
Data Set5	1	3

図６９

【図７０】

Data Set	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
Data Set1	2	2	2	2	2	2	2	2	2	2	2
Data Set2	0	2	2	2	2	2	2	2	2	2	2
Data Set3	0	2	2	2	2	0	2	1	2	1	1
Data Set4	0	2	0	1	1	0	1	1	0	1	1
Data Set5	0	0	0	0	0	0	0	0	0	0	0

履歴情報の項目の組合せ

図７０

【図７１】

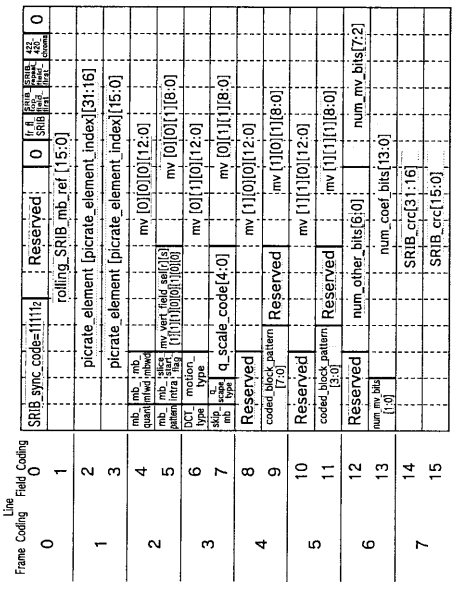


図７１

【図７２】

picture rate elements(72-1)

parameter	number format	number of bits	bit offset from	bit offset to	data category	details
MPEG standard flag	1bit flag	1	0	0	3	1=>MPEG 1;0=>MPEG2
red_bw_flag	1bit flag	1	1	1	3	Default="0"
red_bw_indicator	2bit ui	3	2	4	3	Default="000"
header present flags	2bit flags	2	5	6	3	sequence header present flag, GOP header present flag
Extension start code flags	16 flags	16	7	22	3	Indicates if a given extension start code exists. The 16 flags correspond to the 16 entries in table 6.2 of the ISO/IEC 13818-2: 1996 standard in the order they are listed.
Other start codes	3 flags	3	23	25	3	user_data_start_code, sequence_error_code, sequence_end_code
sequence header						
horizontal_size	14bit uimsbf	14	26	39	2	includes extension
vertical_size	14bit uimsbf	14	40	53	2	includes extension
aspect_ratio_information	4bit uimsbf	4	54	57	2	
frame_rate_code	4bit uimsbf	4	58	61	2	
bit_rate	30bit uimsbf	30	62	91	2	includes extension, correct value should be calculated
vbv_buffer_size	18bit uimsbf	18	92	109	2	includes extension
constrained_parameters_flag	1bit flag	1	110	110	2	
sequence extension						
profile_and_level_indication	8bit uimsbf	8	111	118	2	
progressive_sequence	1bit flag	1	119	119	2	
chroma_format	2bit uimsbf	2	120	121	2	
low_delay	1bit flag	1	122	122	2	
sequence display extension						
video_format	3bit uimsbf	3	123	125	2	
colour_description	1bit flag	1	126	126	2	
colour_primaries	8bit uimsbf	8	127	134	2	
transfer_characteristics	8bit uimsbf	8	135	142	2	
matrix_coefficients	8bit uimsbf	8	143	150	2	
display_horizontal_size	14bit uimsbf	14	151	164	2	
display_vertical_size	14bit uimsbf	14	165	178	2	
group of pictures header						
time_code	25bit flag	25	179	203	2	
closed_gop	1bit flag	1	204	204	2	

図７２

【図７３】

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 bitstream
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	

図７３

【図７４】

picture rate elements(72-3)

frame_centre_vertical_offset_2	16bit uimsbf	16	2393	2408	2	
frame_centre_horizontal_offset_3	16bit uimsbf	16	2409	2424	2	
frame_centre_vertical_offset_3	16bit uimsbf	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 uimsbf	64	2451	2514	2	
PTS/DTS						
PTS_DTS_flag	2bit flag	2	2515	2516	1	
PTS_Value	33bit uimsbf	33	2517	2549	2	
DTS_Value	33bit uimsbf	33	2550	2582	2	
spare reserved bits						
spare	41bit uimsbf	41	2583	2623		
user data area						
user_data		1664	2624	4287	2	
picture rate information CRC						
32-bit_protection_CRT	32bit uimsbf	32	4288	4319		

図７４

【図７５】

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 Bus	Y[0][0]	Embedded Aligned MPEG-2 Re-Coding Information Bus	Y[1][0]	Embedded Aligned MPEG-2 Re-Coding Information Bus	Y[2][0]	Embedded Aligned MPEG-2 Re-Coding Information Bus	Y[3][0]

図７５

【図７６】

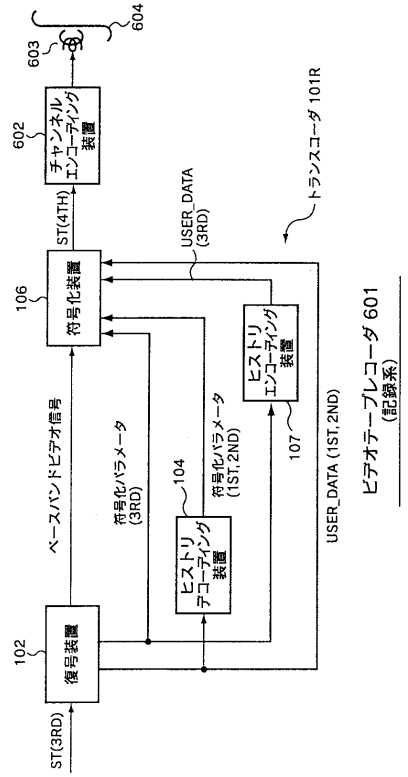


図７６

【図７７】

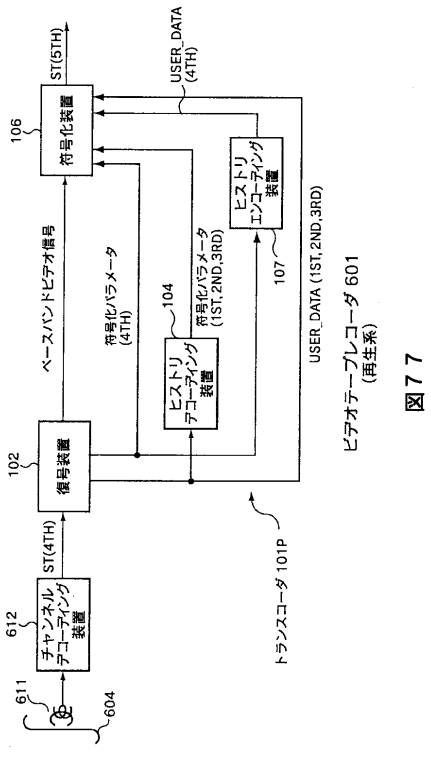


図７７

【図７８】

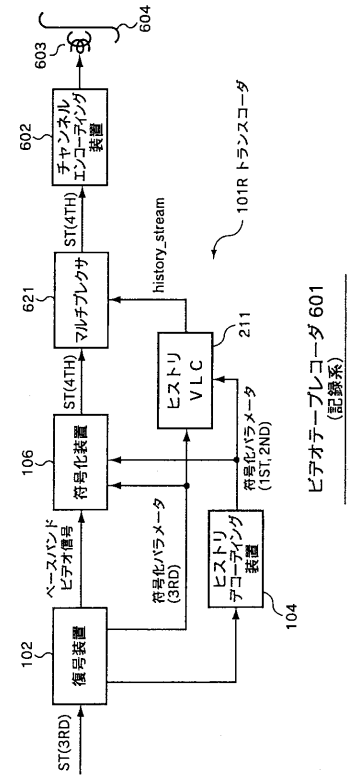


図７８

【図７９】

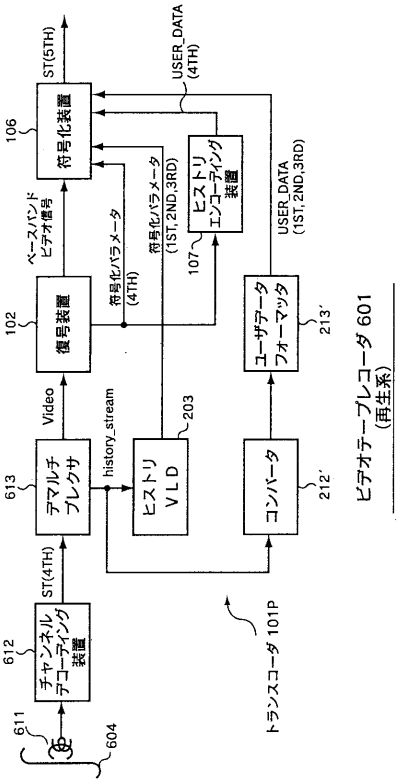


図７９

【図８０】

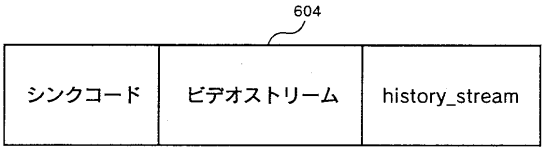


図８０

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
Ｈ０４Ｎ　５／９１　－　５／９５６