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

特許第4676139号
(P4676139)

(45)発行日　平成23年4月27日(2011.4.27)　(24)登録日　平成23年2月4日(2011.2.4)

(51)Int.Cl.　F I
G10L 19/00 (2006.01)　G10L 19/00 213
G10L 19/02 (2006.01)　G10L 19/02 110A
G10L 19/02 150

請求項の数 24　外国語出願　(全 65 頁)

(21)出願番号　特願2003-309276 (P2003-309276)
(22)出願日　平成15年9月1日 (2003.9.1)
(65)公開番号　特開2004-264810 (P2004-264810A)
(43)公開日　平成16年9月24日 (2004.9.24)
審査請求日　平成18年8月23日 (2006.8.23)
(31)優先権主張番号　60/408,517
(32)優先日　平成14年9月4日 (2002.9.4)
(33)優先権主張国　米国 (US)
(31)優先権主張番号　10/642,550
(32)優先日　平成15年8月15日 (2003.8.15)
(33)優先権主張国　米国 (US)

(73)特許権者　500046438
マイクロソフト　コーポレーション
アメリカ合衆国　ワシントン州　９８０５２－６３９９　レッドモンド　ワン　マイクロソフト　ウェイ
(74)代理人　100077481
弁理士　谷　義一
(74)代理人　100088915
弁理士　阿部　和夫
(72)発明者　ナビーン　サンプディ
アメリカ合衆国　９８０７４　ワシントン州　サマミッシュ　ノースイースト　２３４　プレイス　５１８

最終頁に続く

(54)【発明の名称】マルチチャネルオーディオのエンコーディングおよびデコーディング

(57)【特許請求の範囲】

【請求項１】

オーディオエンコーダにおいて、コンピュータにより実施される方法であって、

マルチチャネルオーディオデータを受け取ることと、

マルチチャネル変換の複数の使用可能なタイプの中からマルチチャネル変換を選択することであって、前記マルチチャネル変換の複数の使用可能なタイプの各々は、複数のチャネルにまたがる異なる変換を、前記複数のチャネル中の同一位置にあるウィンドウに対して指定することと、

複数の周波数帯域で前記選択された変換を選択的にオン／オフにすることと、

前記複数の周波数帯域のうちで前記選択された変換がオンである１つまたは複数の周波数帯域で前記オーディオデータに対して前記選択された変換を実行することであって、ここで前記エンコーダが、前記複数の周波数帯域のうちで前記選択された変換がオフである１つまたは複数の周波数帯域で前記オーディオデータに対して変換を実行しないか恒等変換を実行することと

を含むことを特徴とする方法。

【請求項２】

前記複数の周波数帯域ごとに１ビットを含むマスクを出力することをさらに含むことを特徴とする請求項１に記載の方法。

【請求項３】

単一ビットと、前記選択された変換が前記複数の周波数帯域のすべてでオンにされない

場合に前記複数の周波数帯域のそれぞれについて１ビットを含むマスクとを出力することをさらに含むことを特徴とする請求項１に記載の方法。

【請求項４】

前記エンコーダは、前記複数の周波数帯域でのチャネル相関測定値に少なくとも部分的に基づいて前記選択された変換を選択的にオン／オフにすることを特徴とする請求項１に記載の方法。

【請求項５】

オーディオデコーダにおいて、コンピュータにより実施される方法であって、

エンコードされたマルチチャネルオーディオデータを受け取ることと、

逆マルチチャネル変換の複数の使用可能なタイプの中から逆マルチチャネル変換を選択することであって、前記逆マルチチャネル変換の複数の使用可能なタイプの各々は、複数のチャネルにまたがる異なる変換を、前記複数のチャネル中の同一位置にあるウィンドウに対して指定することと、

複数の周波数帯域について周波数帯域オン／オフ選択に関する情報を検索することと、

前記複数の周波数帯域のうちで前記選択された変換がオンである１つまたは複数の周波数帯域で前記オーディオデータに対して前記選択された変換を実行することであって、ここで前記デコーダが、前記複数の周波数帯域のうちで前記選択された変換がオフである１つまたは複数の周波数帯域で前記オーディオデータに対して変換を実行しないか恒等変換を実行することと

を含むことを特徴とする方法。

【請求項６】

前記検索された情報は、前記複数の周波数帯域のそれぞれについて１ビットを含むマスクを含むことを特徴とする請求項５に記載の方法。

【請求項７】

前記検索された情報は、単一ビットと、前記選択された変換が前記複数の周波数帯域のすべてでオンにされない場合に前記複数の周波数帯域のそれぞれについて１ビットを含むマスクとを含むことを特徴とする請求項５に記載の方法。

【請求項８】

オーディオエンコーダにおいて、コンピュータにより実施される方法であって、

マルチチャネルオーディオデータを受け取ることと、

マルチチャネル変換の複数の使用可能なタイプの中からマルチチャネル変換を選択することであって、ここで前記複数の使用可能なタイプが、３つ以上の事前定義の変換を含み、および前記マルチチャネル変換の複数の使用可能なタイプの各々は、複数のチャネルにまたがる異なる変換を、前記複数のチャネル中の同一位置にあるウィンドウに対して指定することと、

前記オーディオデータに対して前記選択された変換を実行することと

を含むことを特徴とする方法。

【請求項９】

前記複数の使用可能なタイプは、さらに、一般的なユニタリ変換を含むことを特徴とする請求項８に記載の方法。

【請求項１０】

前記選択された変換を示す情報を出力することをさらに含むことを特徴とする請求項８に記載の方法。

【請求項１１】

オーディオエンコーダにおいて、コンピュータにより実施される方法であって、

マルチチャネルオーディオデータを受け取ることと、

マルチチャネル変換の複数の使用可能なタイプの中からマルチチャネル変換を選択することであって、ここで前記複数の使用可能なタイプが、複数の事前定義の変換および少なくとも１つのカスタム変換を含み、および前記マルチチャネル変換の複数の使用可能なタイプの各々は、複数のチャネルにまたがる異なる変換を、前記複数のチャネル中の同一位

置にあるウィンドウに対して指定することと、
前記オーディオデータに対して前記選択された変換を実行することと
を含むことを特徴とする方法。
【請求項１２】
前記選択された変換を示す情報を出力することをさらに含むことを特徴とする請求項１１に記載の方法。
【請求項１３】
前記出力される情報は、前記選択された変換の個々の要素に関する情報を含むことを特徴とする請求項１２に記載の方法。
【請求項１４】
前記選択された事前定義の変換の性能は、冗長性除去に関して前記カスタム変換の性能に適当に近い場合に、前記エンコーダは、前記複数の事前定義の変換の１つを選択することを特徴とする請求項１１に記載の方法。
【請求項１５】
オーディオデコーダにおいて、コンピュータにより実施される方法であって、
エンコードされたマルチチャネルオーディオデータを受け取ることと、
逆マルチチャネル変換の複数の使用可能なタイプの中から逆マルチチャネル変換を選択することであって、ここで前記複数の使用可能なタイプが、３つ以上の事前定義の変換を含み、および前記逆マルチチャネル変換の複数の使用可能なタイプの各々は、複数のチャネルにまたがる異なる変換を、前記複数のチャネル中の同一位置にあるウィンドウに対して指定することと、
前記オーディオデータに対して前記選択された変換を実行することと
を含むことを特徴とする方法。
【請求項１６】
前記事前定義の変換は、ＤＣＴ変形形態変換およびアダマール変換を含むことを特徴とする請求項８または１５に記載の方法。
【請求項１７】
前記選択の前に、前記選択された変換を示す情報を検索することをさらに含むことを特徴とする請求項１５に記載の方法。
【請求項１８】
前記複数の使用可能なタイプは、さらに、カスタム変換を含むことを特徴とし、前記検索される情報は、前記カスタム変換を選択するための１つまたは複数の信号を含むことを特徴とし、前記検索される情報は、前記カスタム変換の個々の要素に関する情報をさらに含むことを特徴とする請求項１７に記載の方法。
【請求項１９】
オーディオデコーダにおいて、コンピュータにより実施される方法であって、
エンコードされたマルチチャネルオーディオデータを受け取ることと、
逆マルチチャネル変換の複数の使用可能なタイプの中から逆マルチチャネル変換を選択することであって、ここで前記複数の使用可能なタイプが、複数の事前定義の変換および少なくとも１つのカスタム変換を含み、および前記逆マルチチャネル変換の複数の使用可能なタイプの各々は、複数のチャネルにまたがる異なる変換を、前記複数のチャネル中の同一位置にあるウィンドウに対して指定することと、
前記オーディオデータに対して前記選択された変換を実行することと
を含むことを特徴とする方法。
【請求項２０】
前記選択の前に、前記選択された変換を示す情報を検索することをさらに含むことを特徴とする請求項１９に記載の方法。
【請求項２１】
前記検索される情報は、前記カスタム変換を選択するための１つまたは複数の信号を含み、特に前記検索される情報は、前記カスタム変換の個々の要素に関する情報をさらに含

むことを特徴とする請求項２０に記載の方法。
【請求項２２】
前記マルチチャネルオーディオデータは、２つのチャネルになっていることを特徴とする請求項１，５，８，１１，１５および１９のいずれかの項に記載の方法。
【請求項２３】
前記マルチチャネルオーディオデータは、２つを超えるチャネルになっていることを特徴とする請求項１，５，８，１１，１５および１９のいずれかの項に記載の方法。
【請求項２４】
コンピュータに請求項１，５，８，１１，１５および１９のいずれかの項に記載の方法を実行させるためのプログラムを記録したことを特徴とするコンピュータ可読記録媒体。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、エンコーディングおよびデコーディングでのマルチチャネルオーディオ情報の処理に関する。
【背景技術】
【０００２】
コンパクトディスク、ディジタル無線電話網、およびインターネットを介するオーディオ配信の導入に伴って、ディジタルオーディオがありふれたものになってきた。技術者は、さまざまな手法を使用して、ディジタルオーディオの品質を維持しながら、ディジタルオーディオを効率的に処理する。これらの手法を理解するために、コンピュータでオーディオ情報がどのように表現され、処理されるかを理解することが役に立つ。
【０００３】
Ｉ．コンピュータでのオーディオ情報の表現
コンピュータでは、オーディオ情報を、オーディオ情報を表す一連の数として処理する。たとえば、単一の数が、オーディオサンプルを表すことができ、このオーディオサンプルは、特定の時刻の振幅値（すなわち、音の大きさ）である。サンプル深さ（ｓａｍｐｌｅ　ｄｅｐｔｈ）、サンプリングレート（ｓａｍｐｌｉｎｇ　ｒａｔｅ）、およびチャネルモードなど、複数の要因が、オーディオ情報の品質に影響する。
【０００４】
サンプル深さ（または精度）は、サンプルを表すのに使用される数の範囲を示す。サンプルに可能な値が多くなれば、振幅のより微妙な変動を取り込めるようになるので、品質が高くなる。たとえば、８ビットサンプルは、２５６個の可能な値を有するが、１６ビットサンプルは、６５５３６個の可能な値を有する。２４ビットサンプルでは、普通の音の大きさの変動を非常に微細に取り込むことができ、異常に大きい音も取り込むことができる。
【０００５】
サンプリングレート（通常は、サンプル数毎秒として測定される）も、品質に影響する。サンプリングレートが高いほど、より高い周波数の音を表現できるので、品質が高くなる。一般的なサンプリングレートは、８０００、１１０２５、２２０５０、３２０００、４４１００、４８０００、および９６０００サンプル毎秒である。
【０００６】
モノラルおよびステレオが、オーディオの２つの一般的なチャネルモードである。モノラルモードでは、オーディオ情報が、１つのチャネルに存在する。ステレオモードでは、オーディオ情報が、通常は左チャネルおよび右チャネルと称する２つのチャネルに存在する。５．１チャネル、７．１チャネル、または９．１チャネルのサラウンドサウンド（「１」は、サブウーファ（副低音スピーカ）または低周波数効果チャネルを示す）などの、より多くのチャネルを有する他のモードも可能である。表１に、対応する生ビットレートコストと共に、異なる品質レベルのオーディオの複数のフォーマットを示す。
【０００７】

【表１】

品質	サンプル深さ（ビット／サンプル）	サンプリングレート（サンプル／秒）	モード	生ビットレート（ビット／秒）
インターネット電話	8	8,000	モノラル	64,000
電話	8	11,025	モノラル	88,200
ＣＤオーディオ	16	44,100	ステレオ	1,411,200

表１：異なる品質のオーディオ情報のビットレート

【０００８】

サラウンドサウンドオーディオは、通常は、さらに高い生ビットレートを有する。表１からわかるように、高品質オーディオ情報のコストは、高いビットレートである。高品質オーディオ情報は、大量のコンピュータストレージ（記憶装置）および伝送容量を消費する。しかし、企業および消費者は、高品質マルチチャネルオーディオコンテンツを作成し、配布し、再生するのに、ますますコンピュータに頼る。

【０００９】

ＩＩ．コンピュータでのオーディオ情報の処理

多くのコンピュータおよびコンピュータネットワークに、生ディジタルオーディオを処理するリソース（資源）が不足している。圧縮（エンコーディングまたはコーディングとも称する）では、情報をよりビットレートの低い形に変換することによって、オーディオ情報の保管および伝送のコストを下げる。圧縮は、ロスレス（損失のない）（ｌｏｓｓｌｅｓｓ）（品質に影響しない）、またはロッシイ（損失のある）（ｌｏｓｓｙ）（品質に影響するが、後続のロスレス圧縮からのビットレート削減がより劇的である）とすることができる。圧縮解除（復元）（デコーディング（複合化、解読、逆符号化）とも称する）によって、圧縮された形から元の情報の再構成されたバージョンが抽出される。

【００１０】

Ａ．標準的な知覚オーディオエンコーダおよびデコーダ

一般に、オーディオ圧縮の目標は、オーディオ信号をディジタルに表現して、可能な最小限の量のビットで最高の信号品質をもたらすことである。通常のオーディオエンコーダ／デコーダ［「コーデック（ｃｏｄｅｃ）」］システムでは、サブバンド／変換コーディング、量子化、レート制御、および可変長コーディングを使用して、その圧縮を達成する。量子化および他のロッシイ圧縮手法によって、潜在的に可聴の雑音がオーディオ信号に導入される。雑音の可聴性は、どれほどの雑音があるかと、雑音のどれだけを聴取者が知覚するかに依存する。第１の要因は、主に客観的な品質に関し、第２の要因は、人間による音の知覚に依存する。

【００１１】

図１に、従来技術による変換ベースの知覚オーディオエンコーダ（１００）の一般化された図を示す。図２に、従来技術による対応するオーディオデコーダ（２００）の一般化された図を示す。図１および２に示されたコーデックシステムは、一般化されているが、Ｍｉｃｒｏｓｏｆｔ Ｃｏｒｐｏｒａｔｉｏｎ社のＷｉｎｄｏｗｓ（登録商標）Ｍｅｄｉａ Ａｕｄｉｏ［「WMA」］エンコーダおよびデコーダの諸バージョンを含む、複数の実世界のコーデックシステムに見られる特性を有する。他のコーデックシステムは、Ｍｏｔｉｏｎ Ｐｉｃｔｕｒｅ Ｅｘｐｅｒｔｓ Ｇｒｏｕｐ、Ａｕｄｉｏ Ｌａｙｅｒ ３［「ＭＰ３」］標準規格、Ｍｏｔｉｏｎ Ｐｉｃｔｕｒｅ Ｅｘｐｅｒｔｓ Ｇｒｏｕｐ ２、Ａｄｖａｎｃｅｄ Ａｕｄｉｏ Ｃｏｄｉｎｇ［「ＡＡＣ」］標準規格、およびＤｏｌｂｙ ＡＣ３によって提供されるか指定される。コーデックシステムに関する追加情報については、めいめいの標準規格または技術的刊行物を参照されたい。

【００１２】

１．知覚オーディオエンコーダ

全体として、エンコーダ（１００）は、入力オーディオサンプル（１０５）の時系列を

受け取り、オーディオサンプル（１０５）を圧縮し、エンコーダ（１００）のさまざまなモジュールによって作られる情報を多重化して、ビットストリーム（１９５）を出力する。エンコーダ（１００）には、周波数トランスフォーマ（変換器）（ｆｒｅｑｕｅｎｃｙ　ｔｒａｎｓｆｏｒｍｅｒ）（１１０）、マルチチャネルトランスフォーマ（ｍｕｌｔｉ－ｃｈａｎｎｅｌ　ｔｒａｎｓｆｏｒｍｅｒ）（１２０）、知覚モデラ（モデル信号発生器）（ｐｅｒｃｅｐｔｉｏｎ　ｍｏｄｅｌｅｒ）（１３０）、ウェイタ（重み付け器）（ｗｅｉｇｈｔｅｒ）（１４０）、クォンタイザ（量子化器）（ｑｕａｎｔｉｚｅｒ）（１５０）、エントロピエンコーダ（ｅｎｔｒｏｐｙ　ｅｎｃｏｄｅｒ）（１６０）、コントローラ（１７０）、およびビットストリームマルチプレクサ（ｂｉｔｓｔｒｅａｍ　ｍｕｌｔｉｐｌｅｘｅｒ）［「ＭＵＸ」］（１８０）が含まれる。

【００１３】

周波数トランスフォーマ（１１０）は、オーディオサンプル（１０５）を受け取り、周波数領域のデータに変換する。たとえば、周波数トランスフォーマ（１１０）は、オーディオサンプル（１０５）をブロックに分割し、このブロックは、可変時間分解能を可能にするために可変サイズを有することができる。小さいブロックを用いると、入力オーディオサンプル（１０５）の短いがアクティブな推移セグメント（区間）で時間詳細をより多く保存できるようになるが、ある程度周波数分解能が犠牲になる。対照的に、大きいブロックは、よりよい周波数分解能とより悪い時間分解能を有し、通常は、より長いが少数のアクティブセグメント（区間）でのより高い圧縮効率が可能になる。ブロックをオーバーラップさせて、そうでなければ後の量子化によって導入される可能性があるブロックの間の知覚的不連続性を減らすことができる。マルチチャネルオーディオについて、周波数トランスフォーマ（１１０）では、特定のフレーム内のチャネルごとに同一のパターンのウィンドウが使用される。周波数トランスフォーマ（１１０）は、周波数係数データのブロックをマルチチャネルトランスフォーマ（１２０）に出力し、ブロックサイズなどのサイド情報をＭＵＸ（１８０）に出力する。

【００１４】

マルチチャネルオーディオデータの場合に、周波数トランスフォーマ（１１０）によって作られる周波数係数データの複数のチャネルが、しばしば相関する。この相関を活用するために、マルチチャネルトランスフォーマ（１２０）によって、複数のオリジナルの独立にコーディングされたチャネルを、連繋して（一緒に）コーディングされたチャネルに変換することができる。たとえば、入力がステレオモードである場合に、マルチチャネルトランスフォーマ（１２０）によって、左右のチャネルを和と差のチャネルに変換することができる。

【００１５】

【数１】

$$X_{Sum}[k] = \frac{X_{Left}[k] + X_{Right}[k]}{2} \qquad (1)$$

$$X_{Diff}[k] = \frac{X_{Left}[k] + X_{Right}[k]}{2} \qquad (2)$$

【００１６】

または、マルチチャネルトランスフォーマ（１２０）によって、左右のチャネルを独立にコーディングされたチャネルとして渡すことができる。独立にコーディングされたチャネルまたは連繋してコーディングされたチャネルの使用の判断は、事前に決定するか、エンコーディング中に適応的に行うことができる。たとえば、エンコーダ（１００）によって、（ａ）マルチチャネル変換ありおよびなしのコーディングチャネルの間のエネルギ分離と、（ｂ）左右の入力チャネルの間の励起パターンの不一致を考慮するオープンループ選択判断を用いて、ステレオチャネルを連繋してまたは独立にのどちらでコーディングするかを決定する。そのような判断は、ウィンドウごとの基準で行うか、判断を単純にするためにフレームごとに１回だけ行うことができる。マルチチャネルトランスフォーマ（１２

０）は、使用されるチャネルモードを示すサイド情報をＭＵＸ（１８０）に出力する。

【００１７】

エンコーダ（１００）は、マルチチャネル変換の後に、オーディオデータのブロックにマルチチャネル再行列化を適用することができる。連繋してコーディングされたチャネルの低ビットレートのマルチチャネルオーディオデータについて、エンコーダ（１００）は、あるチャネル（たとえば差チャネル）の情報を選択的に抑圧して、残りのチャネル（たとえば和チャネル）の品質を改善する。たとえば、エンコーダ（１００）は、スケーリング係数ρによって差チャネルをスケーリングする。

【００１８】

【数２】

$$\tilde{X}_{Diff}[k] = \rho \cdot X_{Diff}[k] \qquad (3)$$

【００１９】

ここで、ρの値は、（ａ）雑音興奮比率（Ｎｏｉｓｅ　ｔｏ　Ｅｘｃｉｔａｔｉｏｎ　Ｒａｔｉｏ）［「ＮＥＲ」］などの知覚オーディオ品質測定値の現在の平均レベルと、（ｂ）仮想バッファの現在の満杯度と、（ｃ）エンコーダ（１００）のビットレートおよびサンプリングレート設定と、（ｄ）左右の入力チャネルのチャネルセパレーションとに基づく。

【００２０】

知覚モデラ（１３０）は、人間の聴覚系のモデルに従ってオーディオデータを処理して、所与のビットレートの再構成されたオーディオ信号の知覚される品質を改善する。たとえば、聴覚モデルでは、通常、人間の聴取帯および臨界帯域の範囲が考慮される。人間の神経系では、周波数のサブレンジが統合される。この理由から、聴覚モデルでは、臨界帯域によってオーディオ情報を編成し、処理することができる。異なる聴覚モデルでは、異なる数の臨界帯域（たとえば、２５個、３２個、５５個、または１０９個）および／または臨界帯域の異なるカットオフ周波数が使用される。バークバンド（ｂａｒｋ　ｂａｎｄ：叫び声の帯域）が、臨界帯域の周知の例である。範囲および臨界帯域のほかに、オーディオ信号の間の相互作用が、知覚に劇的に影響する可能性がある。単独で提示される場合に明瞭に聴取可能であるオーディオ信号が、マスカまたはマスキング信号と称する別のオーディオ信号が存在すると完全に聴取不能になる可能性がある。人間の耳は、マスキングされる信号のひずみまたは他の忠実度の消失（すなわち雑音）に比較的鈍感であり、したがって、マスキングされる信号に、知覚されるオーディオ品質を劣化させずにより多くのひずみを含めることができる。さらに、聴覚モデルでは、人間による音の知覚の物理的態様または神経的態様に関するさまざまな他の要因を考慮することができる。

【００２１】

知覚モデラ（１３０）は、雑音の可聴性を減らすためにオーディオデータの雑音を整形するのにウェイタ（１４０）が使用する情報を出力する。たとえば、さまざまな手法のいずれかを使用して、ウェイタ（１４０）は、受け取った情報に基づいて量子化行列（時々、マスクと称する）の重みづけ係数（時々、スケーリング係数と称する）を生成する。量子化行列の重みづけ係数には、オーディオデータ内の複数の量子化帯域ごとの重みが含まれ、量子化帯域は、周波数係数の周波数範囲である。量子化帯域の数は、臨界帯域の数以下とすることができる。したがって、重みづけ係数によって、雑音が量子化帯域にまたがって分散する特性が示され、より多くの雑音をより聴取可能でない帯域に置き、より少ない雑音をより聴取可能な帯域に置くことによって、雑音の可聴性を最小にすることが目標になる。重みづけ係数は、振幅およびブロックからブロックへの量子化帯域の数で変化する可能性がある。ウェイタ（１４０）は、マルチチャネルトランスフォーマ（１２０）から受け取ったデータに重みづけ係数を適用する。

【００２２】

一実施形態で、ウェイタ（１４０）は、マルチチャネルオーディオの各チャネルのウィンドウごとに重みづけ係数の組を生成するか、連繋してコーディングされたチャネルの並

列ウィンドウについて重みづけ係数の単一の組を共用する。ウェイタ（１４０）は、係数データの重みづけされたブロックをクォンタイザ（１５０）に出力し、重みづけ係数の組などのサイド情報をＭＵＸ（１８０）に出力する。

【００２３】

重みづけ係数の組を、直接圧縮を使用して、より効率的な表現のために圧縮することができる。直接圧縮手法では、エンコーダ（１００）が、量子化行列の各要素を均一に量子化する。エンコーダは、量子化された要素を、行列の前の要素に対して相対的に差分コーディングし、差分コーディングされた要素をハフマンコーディングする。いくつかの場合に（たとえば、特定の量子化帯域の係数のすべてが、０の値に量子化されるか切り詰められる時）、デコーダ（２００）は、すべての量子化帯域について重みづけ係数を必要としない。そのような場合には、エンコーダ（１００）は、系列の次に必要な重みづけ係数の値と同一の値を、１つまたは複数の不要な重みづけ係数に与え、これによって、量子化行列の要素の差分コーディングをより効率的にする。

【００２４】

あるいは、低ビットレートアプリケーションについて、エンコーダ（１００）は、量子化行列をパラメータ圧縮して、たとえば量子化行列から計算される擬似自己相関パラメータの線形予測コーディング［「ＬＰＣ」］を使用して、パラメータの組として量子化行列を表現することができる。

【００２５】

クォンタイザ（１５０）は、ウェイタ（１４０）の出力を量子化し、エントロピエンコーダ（１６０）への量子化された係数データと、ＭＵＸ（１８０）への量子化ステップサイズを含むサイド情報とを作る。量子化では、入力値の範囲を単一の値に写像し、情報の不可逆的な消失が導入されるが、量子化によって、エンコーダ（１００）が、コントローラ（１７０）と共に、ビットストリーム（１９５）出力の品質およびビットレートを調整できるようになる。図１では、クォンタイザ（１５０）が、適応均一スカラクォンタイザである。クォンタイザ（１５０）は、各周波数係数に同一の量子化ステップサイズを適用するが、量子化ステップサイズ自体を、量子化ループのある反復から次の反復の間で変更して、エントロピエンコーダ（１６０）出力のビットレートに影響を及ぼすことができる。他の種類の量子化が、不均一ベクトル量子化および／または非適応量子化である。

【００２６】

エントロピエンコーダ（１６０）は、クォンタイザ（１５０）から受け取る量子化された係数データをロスレス圧縮する。エントロピエンコーダ（１６０）は、オーディオ情報のエンコーディングに費やされるビット数を計算し、この情報をレート／品質コントローラ（１７０）に渡すことができる。

【００２７】

コントローラ（１７０）は、クォンタイザ（１５０）と一緒に働いて、エンコーダ（１００）の出力のビットレートおよび／または品質を調整する。コントローラ（１７０）は、エンコーダ（１００）の他のモジュールから情報を受け取り、受け取った情報を処理して、現在の条件での所望の量子化ステップサイズを判定する。コントローラ（１７０）は、ビットレート制約および品質制約を満足することを目標に、量子化ステップサイズをクォンタイザ（１５０）に出力する。

【００２８】

エンコーダ（１００）は、オーディオデータのブロックに雑音置換および／または帯域切詰を適用することができる。低ビットレートおよび中ビットレートで、オーディオエンコーダ（１００）は、雑音置換を使用して、ある帯域の情報を伝える。帯域切詰では、ブロックの測定された品質から低い品質が示される場合に、エンコーダ（１００）が、ある（通常はより高い周波数の）帯域の係数を完全に除去して、残りの帯域の総合的な品質を改善することができる。

【００２９】

ＭＵＸ（１８０）は、オーディオエンコーダ（１００）の他のモジュールから受け取る

サイド情報を、エントロピエンコーダ（１６０）から受け取ったエントロピエンコーディングされたデータと多重化する。ＭＵＸ（１８０）は、オーディオデコーダが認識するフォーマットで情報を出力する。ＭＵＸ（１８０）には、オーディオの複雑さの変化に起因するビットレートの短期間変動を平滑化するために、エンコーダ（１００）によって出力されるビットストリーム（１９５）を保管する仮想バッファが含まれる。

【００３０】

２．知覚オーディオデコーダ

全体として、デコーダ（２００）は、エントロピエンコードされたデータならびにサイド情報を含む圧縮オーディオ情報のビットストリーム（２０５）を受け取り、このビットストリームから、オーディオサンプル（２９５）を再構成する。オーディオデコーダ（２００）には、ビットストリームデマルチプレクサ［「ＤＥＭＵＸ」］（２１０）、エントロピデコーダ（２２０）、逆クォンタイザ（２３０）、雑音ジェネレータ（２４０）、逆ウェイタ（２５０）、逆マルチチャネルトランスフォーマ（２６０）、および逆周波数トランスフォーマ（２７０）が含まれる。

【００３１】

ＤＥＭＵＸ（２１０）は、ビットストリーム（２０５）の情報を解析し、情報をデコーダ（２００）のモジュールに送る。ＤＥＭＵＸ（２１０）には、オーディオの複雑さの変動、ネットワークジッタ、および／または他の要因に起因するビットレートの短期間変動を補償するために、１つまたは複数のバッファが含まれる。

【００３２】

エントロピデコーダ（２２０）は、ＤＥＭＵＸ（２１０）から受け取ったエントロピコードをロスレス圧縮解除し、量子化された周波数係数データを作る。エントロピデコーダ（２２０）は、通常は、エンコーダで使用されるエントロピエンコーディング手法の逆を適用する。

【００３３】

逆クォンタイザ（２３０）は、ＤＥＭＵＸ（２１０）から量子化ステップサイズを受け取り、エントロピデコーダ（２２０）から量子化された周波数係数データを受け取る。逆クォンタイザ（２３０）は、量子化された周波数係数データに量子化ステップサイズを適用して、周波数係数データを部分的に再構成する。

【００３４】

雑音ジェネレータ（２４０）は、ＤＥＭＵＸ（２１０）から、データのブロックのどの帯域が雑音置換されたかを示す情報と、雑音の形に関するパラメータを受け取る。雑音ジェネレータ（２４０）は、示された帯域のパターンを生成し、その情報を逆ウェイタ（２５０）に渡す。

【００３５】

逆ウェイタ（２５０）は、ＤＥＭＵＸ（２１０）から重みづけ係数を受け取り、雑音ジェネレータ（２４０）から雑音置換された帯域のパターンを受け取り、逆クォンタイザ（２３０）から部分的に再構成された周波数係数データを受け取る。必要に応じて、逆ウェイタ（２５０）は、たとえば、量子化された行列の要素のエントロピデコーディング、逆差分コーディング、および逆量子化などによって、重みづけ係数を圧縮解除する。逆ウェイタ（２５０）は、雑音置換されなかった帯域の部分的に再構成された周波数係数データに、重みづけ係数を適用する。その後、逆ウェイタ（２５０）は、雑音置換された帯域に関する雑音ジェネレータ（２４０）から受け取った雑音パターンを加える。

【００３６】

逆マルチチャネルトランスフォーマ（２６０）は、逆ウェイタ（２５０）から再構成された周波数係数データを受け取り、ＤＥＭＵＸ（２１０）からチャネルモード情報を受け取る。マルチチャネルオーディオが、独立にコーディングされたチャネルにある場合には、逆マルチチャネルトランスフォーマ（２６０）は、チャネルをそのまま通す。マルチチャネルデータが、連繋してコーディングされたチャネルにある場合には、逆マルチチャネルトランスフォーマ（２６０）は、そのデータを独立にコーディングされたチャネルに変

換する。

【００３７】

逆周波数トランスフォーマ（２７０）は、マルチチャネルトランスフォーマ（２６０）によって出力された周波数係数データならびにＤＥＭＵＸ（２１０）からのブロックサイズなどのサイド情報を受け取る。逆周波数トランスフォーマ（２７０）は、エンコーダで使用される周波数変換の逆を適用し、再構成されたオーディオサンプル（２９５）のブロックを出力する。

【００３８】

【非特許文献１】Yang et al., "An Inter-Channel Redundancy Removal Approach for High-Quality Multichannel Audio Compression," AES 109th Convention, Los Angeles, September 2000 ["Yang"]

【非特許文献２】Wang et al., "A Multichannel Audio Coding Algorithm for Inter-Channel Redundancy Removal," AES 110th Convention, Amsterdam, Netherlands, May 2001 ["Wang"]

【非特許文献３】Kuo et al, "A Study of Why Cross Channel Prediction Is Not Applicable to Perceptual Audio Coding," IEEE Signal Proc. Letters, vol. 8, no. 9, September 2001

【非特許文献４】Rao et al., Discrete Cosine Transform, Academic Press (1990)

【非特許文献５】Vaidyanathan, Multirate Systems and Filter Banks, Chapter 14.6, "Factorization of Unitary Matrices," Prentice Hall (1993)

【発明の開示】

【発明が解決しようとする課題】

【００３９】

Ｂ．標準的な知覚オーディオエンコーダおよび知覚オーディオデコーダの短所

上で説明した知覚エンコーダおよび知覚デコーダは、多くの応用例について良好な総合性能を有するが、複数の短所、特にマルチチャネルオーディオの圧縮および圧縮解除に関する短所を有する。この短所によって、いくつかの場合、たとえば使用可能なビットレートが、入力オーディオチャネルの数に対して少ない時に、再構成されるマルチチャネルオーディオの品質が制限される。

【００４０】

１．マルチチャネルオーディオのフレーム区分での柔軟性のなさ

さまざまな点で、図１のエンコーダ（１００）によって実行されるフレーム区分は、柔軟でない。

【００４１】

前に述べたように、周波数トランスフォーマ（１１０）は、入力オーディオサンプル（１０５）のフレームを、周波数変換のために１つまたは複数のオーバーラップするウィンドウに分割するが、大きいウィンドウは、よりよい周波数分解能および冗長性除去をもたらし、小さいウィンドウは、よりよい時間分解能をもたらす。よりよい時間分解能は、信号が低エネルギから高エネルギに推移する時に導入される可聴プリエコー（ｐｒｅ－ｅｃｈｏ）アーチファクトを制御するのに役立つが、小さいウィンドウを使用すると、圧縮可能性が下がるので、エンコーダは、ウィンドウサイズを選択する時に、これらの考慮事項のバランスをとらなければならない。マルチチャネルオーディオについて、周波数トランスフォーマ（１１０）は、フレームのチャネルを同一の形で（すなわち、チャネルでの同一のウィンドウ構成）区分するが、これは、図３ａから３ｃに示されているように、いくつかの場合に非効率的である場合がある。

【００４２】

図３ａに、例のステレオオーディオ信号の波形（３００）を示す。チャネル０の信号には、推移アクティビティが含まれ、チャネル１の信号は、相対的に静止している。エンコーダ（１００）は、チャネル０の信号推移を検出し、プリエコーを減らすために、フレームを、図３ｂに示された、より小さいオーバーラップする変調されたウィンドウ（３０１

）に分割する。図を単純にするために、図３ｃでは、オーバーラップするウィンドウ構成（３０２）をボックス（箱）で示し、破線によってフレーム境界を示す。後の図も、この規約に従う。

【００４３】

すべてのチャネルに同一のウィンドウ構成をとらせることの短所は、１つまたは複数のチャネルの静止信号（たとえば図３ａから３ｃのチャネル１）が、より小さいウィンドウに分割され、コーディング利得が下がる可能性があることである。その代わりに、エンコーダ（１００）が、すべてのチャネルにより長いウィンドウを使用させることができるが、推移を有する１つまたは複数のチャネルにプリエコーが導入される。この問題は、複数のチャネルをコーディングしなければならない時に悪化する。

【００４４】

ＡＡＣ（適応オーディオコーディング）を用いると、マルチチャネル変換の対単位のチャネルのグループ化が可能になる。左、右、中央、左後ろ、右後ろのチャネルの中から、たとえば、左チャネルと右チャネルをステレオコーディングのためにグループ化し、左後ろチャネルと右後ろチャネルをステレオコーディングのためにグループ化することができる。異なるグループが、異なるウィンドウ構成を有することができるが、特定のグループの両方のチャネルが、ステレオコーディングが使用される場合に同一のウィンドウ構成を有する。これによって、ＡＡＣシステムでのマルチチャネル変換の区分の柔軟性が制限され、対単位のみでのグループ化の使用についても同様である。

【００４５】

２．マルチチャネル変換での柔軟性のなさ

図１のエンコーダ（１００）では、あるチャネル間冗長性（ｉｎｔｅｒ－ｃｈａｎｎｅｌ　ｒｅｄｕｎｄａｎｃｙ）が活用されるが、マルチチャネル変換に関するさまざまな点で柔軟性がない。エンコーダ（１００）を用いると、２種類の変換すなわち、（ａ）恒等変換（変換なしと同等である）、または（ｂ）ステレオ対の和－差コーディングが可能である。これらの制限によって、３つ以上のチャネルのマルチチャネルコーディングが制約される。３つ以上のチャネルを扱うことができるＡＡＣにおいても、マルチチャネル変換は、１時に１対のチャネルだけに制限される。

【００４６】

複数のグループが、サラウンドサウンドチャネルに関するマルチチャネル変換に関して実験した（たとえば、非特許文献１（以下「Ｙａｎｇ」）、非特許文献２（以下「Ｗａｎｇ」）参照）。Ｙａｎｇのシステムでは、よい圧縮係数に関してチャネルを相関解除（ｄｅｃｏｒｒｅｌａｔｅ）するために、チャネルにまたがるＫａｒｈｕｎｅｎ－Ｌｏｅｖｅ変換［「ＫＬＴ」］が使用される。Ｗａｎｇのシステムでは、整数対整数離散コサイン変換（Ｄｉｓｃｒｅｔｅ　Ｃｏｓｉｎｅ　Ｔｒａｎｓｆｏｒｍ）［「ＤＣＴ」］が使用される。両方のシステムで、よい結果が与えられるが、まだ複数の制限がある。

【００４７】

第１に、オーディオサンプルにＫＬＴを使用する（Ｙａｎｇのシステムのように時間領域または周波数領域で）と、再構成で導入されるひずみが制御されない。Ｙａｎｇのシステムの ＫＬＴは、マルチチャネルオーディオの知覚オーディオコーディングに成功裡に使用されない。Ｙａｎｇのシステムでは、逆マルチチャネル変換での、１つの（たとえば激しく量子化される）コーディングされたチャネルから複数の再構成されるチャネルへの漏れの量が制御されない。この短所は、文献で指摘されている（たとえば、非特許文献３参照）。言い換えると、あるコーディングされたチャネルで「可聴でない」量子化が、複数の再構成されたチャネルに分散される時に可聴になる可能性がある。というのは、逆重みづけが、逆マルチチャネル変換の前に実行されるからである。Ｗａｎｇのシステムでは、マルチチャネル変換を、エンコーダ内で重みづけおよび量子化の後に配置する（かつ、逆マルチチャネル変換を、デコーダ内で逆量子化および逆重みづけの前に配置する）ことによって、この問題が克服される。しかし、Ｗａｎｇのシステムは、さまざまな他の短所を有する。マルチチャネル変換の前に量子化を実行することは、マルチチャネル変換を整数

対整数にしなければならず、可能な変換の数が制限され、チャネルにまたがる冗長性除去が制限されることを意味する。

【００４８】

第２に、Ｙａｎｇのシステムは、ＫＬＴ変換に制限される。ＫＬＴ変換は、圧縮されるオーディオデータに適応されるが、Ｙａｎｇのシステムの、異なる種類の変換を使用する柔軟性は、制限されている。同様に、Ｗａｎｇのシステムでは、マルチチャネル変換に整数対整数ＤＣＴが使用されるが、これは、エネルギコンパクト化に関して通常のＤＣＴほど良好ではなく、Ｗａｎｇのシステムの、異なる種類の変換を使用する柔軟性は、制限されている。

【００４９】

第３に、Ｙａｎｇのシステムおよび Ｗａｎｇのシステムには、どのチャネルを一緒に変換するかを制御する機構がなく、マルチチャネル変換の異なる時に異なるチャネルを選択的にグループ化する機構もない。そのような制御は、まったく互換性がないチャネルにまたがるコンテンツの漏れを制限するのに役立つ。さらに、全体的に互換性のあるチャネルであっても、ある期間にわたって互換性がなくなる場合がある。

【００５０】

第４に、Ｙａｎｇのシステムでは、マルチチャネル変換に、周波数帯域レベルでマルチチャネル変換を適用するか否かに対する制御が欠けている。全体的に互換性があるチャネルの間であっても、それらのチャネルが、ある周波数またはある周波数帯域で互換性がない場合がある。同様に、図１のエンコーダ（１００）のマルチチャネル変換には、サブチャネルレベルでの制御が欠けており、どの帯域の周波数係数データをマルチチャネル変換するかが制御されず、入力チャネルの周波数帯域のうちに相関しないものがある時に生じる可能性がある非効率性が無視される。

【００５１】

第５に、ソースチャネルに互換性がある時であっても、しばしば、一緒に変換されるチャネルの数を制御して、変換を実施する間のデータオーバーフローを制限し、メモリアクセスを減らす必要がある。具体的に言うと、Ｙａｎｇのシステムの ＫＬＴは、計算的に複雑である。その一方で、変換サイズを減らすことによって、潜在的に、より大きい変換と比較したコーディング利得も減る。

【００５２】

第６に、マルチチャネル変換を指定する情報を送ることが、ビットレートに関して高コストになる可能性がある。これは、Ｙａｎｇのシステムの ＫＬＴに関して特にそうである。というのは、送られる共分散行列の変換係数が、実数であるからである。

【００５３】

第７に、低ビットレートマルチチャネルオーディオに関して、再構成されるチャネルの品質が、非常に限られる。低ビットレートのコーディングの要件のほかに、これは、部分的に、情報が実際にエンコードされるチャネルの数をシステムが選択的に優雅に削減する能力がないことに起因する。

【００５４】

３．量子化および重みづけの非効率性

図１のエンコーダ（１００）では、ウェイタ（１４０）が、オーディオデータの帯域にまたがるひずみを整形し、クォンタイザ（１５０）が、量子化ステップサイズをセットして、フレームに関するひずみの振幅を変更し、これによって品質とビットレートのバランスをとる。エンコーダ（１００）は、ほとんどの応用例で品質とビットレートのよいバランスを達成するが、エンコーダ（１００）は、まだ複数の短所を有する。

【００５５】

第１に、エンコーダ（１００）には、チャネルレベルでの品質に対する直接制御が欠けている。重みづけ係数によって、個々のチャネルの量子化帯域にまたがる全体的なひずみが整形される。この均一のスカラ量子化ステップサイズは、あるフレームのすべての周波数帯域およびチャネルにまたがるひずみの振幅に影響する。すべてのチャネルでの非常に

高い品質または非常に低い品質の強制がないので、エンコーダ（１００）には、すべてのチャネルの再構成された出力の同等の品質または少なくとも匹敵する品質の設定に対する直接制御が欠けている。

【００５６】

第２に、重みづけ係数がロッシイ圧縮されるので、エンコーダ（１００）には、重みづけ係数の量子化の分解能に対する制御が欠けている。量子化行列の直接圧縮に関して、エンコーダ（１００）は、量子化行列の要素を均一に量子化し、その後、差分コーディングおよびハフマンコーディングを使用する。マスク要素の均一の量子化は、使用可能なビットレートまたは信号の複雑さの変化に適応しない。その結果、量子化行列が、再構成されたオーディオの全体的に低い品質に対して必要以上に高い分解能でエンコードされる場合があり、量子化行列が、再構成されたオーディオの高い品質に対して使用すべき分解能より低い分解能でエンコードされる場合がある。

【００５７】

第３に、エンコーダ（１００）での量子化行列の直接圧縮では、量子化行列の時間的冗長性を活用することができない。直接圧縮では、特定の量子化行列内の冗長性が除去されるが、一連の量子化行列の時間的冗長性が無視される。

【００５８】

Ｃ．オーディオチャネルのダウンミキシング（ｄｏｗｎ－ｍｉｘｉｎｇ）

マルチチャネルオーディオのエンコーディングおよびデコーディングはさておき、Ｄｏｌｂｙ　Ｐｒｏ－Ｌｏｇｉｃおよび複数の他のシステムは、マルチチャネルオーディオのダウンミキシングを実行して、異なる数のスピーカを有するスピーカ構成との互換性を容易にする。Ｄｏｌｂｙ　Ｐｒｏ－Ｌｏｇｉｃのダウンミキシングでは、たとえば、４チャネルが、２チャネルにミックスダウンされ、２チャネルのそれぞれが、元の４つのチャネルのオーディオデータのある組合せを有する。この２チャネルを、ステレオチャネル装置で出力することができ、あるいは、４チャネルを、２チャネルから再構成して、４チャネル機器で出力することができる。

【００５９】

この性質のダウンミキシングによって、互換性問題の一部が解決されるが、これは、あるセット構成、たとえば、４チャネルから２チャネルへのダウンミキシングに制限される。さらに、ミキシングの式が、事前に決定され、信号に適応するための経時的な変化が許容されない。

【課題を解決するための手段】

【００６０】

要約すると、本明細書で詳述する本発明は、マルチチャネルオーディオをエンコードし、デコードする方法を対象とする。たとえば、オーディオエンコーダで、１つまたは複数の手法を使用して、マルチチャネルオーディオデータの品質および／またはビットレートを改善する。これによって、全体的な聴取経験が改善され、コンピュータシステムが、高品質マルチチャネルオーディオの作成、配信、および再生のより説得力のあるプラットフォームになる。本明細書で説明するエンコーディング対策およびデコーディング対策には、さまざまな手法およびツールが含まれ、これらは、組み合わせてまたは独立に使用することができる。

【００６１】

本明細書で説明する本発明の第１の態様によれば、オーディオエンコーダが、マルチチャネルオーディオデータに対して前処理マルチチャネル変換を実行する。エンコーダは、品質を制御するためにエンコーディング中に変換を変更する。たとえば、低ビットレートコーディングについて、エンコーダは、元のオーディオチャネルの１つまたは複数を変更するか捨てて、コーディングの複雑さを減らし、オーディオの全体的な知覚される品質を改善する。

【００６２】

本明細書で説明する本発明の第２の態様によれば、オーディオデコーダが、デコードさ

れたマルチチャネルオーディオデータに対する後処理マルチチャネル変換を実行する。デコーダは、複数の異なる目的のいずれかに関する変換を使用する。たとえば、デコーダは、任意選択として、時間領域オーディオサンプルを再行列化して、再生でのファントムチャネル（ｐｈａｎｔｏｍ　ｃｈａｎｎｅｌ）を作成するか、特殊効果を実行する。

【００６３】

本明細書で説明する本発明の第３の態様によれば、オーディオエンコーダが、異なるチャネルからの複数のウィンドウを１つまたは複数のタイル（ｔｉｌｅ）にグループ化し、タイル構成情報を出力する。たとえば、エンコーダは、ウィンドウが同一の開始時刻および同一の停止時刻を有する時に、異なるチャネルからのウィンドウを単一のタイルにグループ化し、これによって、エンコーダが、小さいウィンドウを有する特定のチャネルに現れる過渡（トランジェント）を分離する（プリエコーアーチファクトを減らす）ことができるが、他のチャネルでの周波数分解能および時間的冗長性削減に大きいウィンドウを使用することができるようになる。

【００６４】

本明細書で説明する本発明の第４の態様によれば、オーディオエンコーダが、マルチチャネルオーディオデータに重みをつけ、その後、重みづけの後だが後の量子化の前に、重みをつけられたオーディオデータに対してマルチチャネル変換を実行する。この順序付けによって、再構成時のチャネルにまたがる可聴量子化雑音の漏れを減らすことができる。

【００６５】

本明細書で説明する本発明の第５の態様によれば、オーディオエンコーダが、オーディオデータの複数のチャネルグループを、マルチチャネル変換のための複数のチャネルグループに選択的にグループ化する。エンコーダは、オーディオシーケンス内の異なる時刻に異なる形で複数のチャネルをグループ化する。これによって、エンコーダにデータの比較的相関する部分へのマルチチャネル変換の適用に対するより正確な制御を与えることによって、性能を改善することができる。

【００６６】

本明細書で説明する本発明の第６の態様によれば、オーディオエンコーダが、複数の周波数帯域での選択された変換を選択的にオン／オフに切り替える。たとえば、エンコーダは、マルチチャネル変換で互換でない帯域を選択的に除外し、やはり、これによって、エンコーダに、データの比較的相関する部分へのマルチチャネル変換の適用に対するより正確な制御が与えられる。

【００６７】

本明細書で説明する本発明の第７の態様によれば、オーディオエンコーダが、複数ステージのマルチチャネル変換の階層に従って、マルチチャネルオーディオデータを変換する。たとえば、この階層によって、他の変換と比較して計算の複雑さを減らしながら別の変換がエミュレートされる。

【００６８】

本明細書で説明する本発明の第８の態様によれば、オーディオエンコーダが、複数の使用可能なタイプのマルチチャネル変換の中からマルチチャネル変換を選択する。たとえば、このタイプには、事前定義の変換およびカスタム変換が含まれる。この形で、エンコーダが、変換を指定するのに使用されるビットレートを減らす。

【００６９】

本明細書で説明する本発明の第９の態様によれば、オーディオエンコーダが、任意のユニタリ変換行列を計算し、その後、その行列を因数分解する。エンコーダは、因数分解された変換を実行し、それに関する情報を出力する。この形で、エンコーダによって、マルチチャネル変換行列が効果的に圧縮される。

【００７０】

オーディオエンコーダに関して上で説明した態様のいくつかに関して、オーディオデコーダによって、対応する処理およびデコーディングが実行される。

【００７１】

本発明のさまざまな特徴および効果は、添付図面に関して進行する、実施形態の以下の詳細な説明から明白になる。

【発明を実施するための最良の形態】

【００７２】

本発明の、説明される実施形態は、エンコーディングおよびデコーディングでオーディオ情報を処理する手法およびツールを対象とする。説明される実施形態では、オーディオエンコーダで、エンコーディング中に、複数の手法を使用してオーディオを処理する。オーディオデコーダでは、デコード中に、複数の手法を使用して、オーディオを処理する。本明細書のところどころで、単一の統合されたシステムの一部として手法を説明するが、これらの手法は、別々に、潜在的には他の手法と組み合わせて、適用することができる。代替実施形態では、エンコーダまたはデコーダ以外のオーディオ処理ツールによって、手法の１つまたは複数が実施される。

【００７３】

いくつかの実施形態で、エンコーダが、マルチチャネル前処理を実行する。低ビットレートコーディングについて、たとえば、エンコーダは、任意選択として、時間領域オーディオサンプルを再行列化して、相互チャネル相関性を人工的に増やす。これによって、コーディングの複雑さを減らすことによって、影響されるチャネルの後続の比較がより効率的になる。前処理によって、チャネルセパレーションが低下するが、全体的な品質を改善することができる。

【００７４】

いくつかの実施形態で、エンコーダおよびデコーダが、ウィンドウのタイルに構成されたマルチチャネルオーディオを扱う。たとえば、エンコーダが、チャネルごとの基準でマルチチャネルオーディオのフレームを区分し、各チャネルが、他のチャネルと独立のウィンドウ構成を有することができるようにする。エンコーダは、区分されたチャネルのウィンドウを、マルチチャネル変換用のタイルにグループ化する。これによって、エンコーダが、小さいウィンドウを有するフレームの特定のチャネルに現れる推移を分離する（プリエコーアーチファクトを減らす）ことができるが、フレームの他のチャネルでの周波数分解能および時間的冗長性削減に大きいウィンドウを使用することができるようになる。

【００７５】

いくつかの実施形態で、エンコーダが、１つまたは複数の柔軟なマルチチャネル変換手法を実行する。デコーダは、対応する逆マルチチャネル変換手法を実行する。第１の手法では、エンコーダが、エンコーダでの知覚的重みづけの後にマルチチャネル変換を実行し、これによって、再構成時のチャネルにまたがる可聴量子化雑音の漏れが減る。第２の手法では、エンコーダが、マルチチャネル変換についてチャネルを柔軟にグループ化して、異なる時にチャネルを選択的に含める。第３の手法では、エンコーダが、柔軟にマルチチャネル変換に特定の周波数帯域を含めるか除外して、互換性のある帯域を選択的に含める。第４の手法では、エンコーダが、選択的に事前定義の行列を使用するか、ギブンス回転を使用してカスタム変換行列をパラメータ化することによって、変化行列に関連するビットレートを減らす。第５の手法では、エンコーダが、柔軟な階層マルチチャネル変換を実行する。

【００７６】

いくつかの実施形態で、エンコーダが、１つまたは複数の改善された量子化手法または改善された重みづけ手法を実行する。対応するデコーダが、対応する逆量子化手法または逆重みづけ手法を実行する。第１の手法では、エンコーダが、チャネルごとの量子化ステップ変更子を計算し、適用し、この変更子によって、エンコーダに、チャネル間の再構成品質のバランスに対するより多くの制御が与えられる。第２の手法では、エンコーダが、量子化行列要素の柔軟な量子化ステップサイズを使用し、これによって、エンコーダが、量子化行列要素の分解能を変更できるようになる。第３の手法では、エンコーダが、量子化行列の圧縮で時間予測を使用して、ビットレートを減らす。

【００７７】

いくつかの実施形態で、デコーダが、マルチチャネル後処理を実行する。たとえば、デコーダが、任意選択として、時間領域オーディオサンプルを再行列化して、再生時にファントムチャネルを作成し、特殊効果を実行し、より少ないスピーカでの再生のためまたは他の目的のためにチャネルを折り畳む。

【００７８】

説明される実施形態では、マルチチャネルオーディオに、図４の行列（４００）に示されているように、標準的な５．１チャネル／スピーカ構成の６チャネルが含まれる。「５」チャネルは、左、右、中央、左後ろ、および右後ろのチャネルであり、サラウンドサウンド用に普通に空間的に配置される。「１」チャネルは、サブウーファまたは低周波数効果チャネルである。説明を明瞭にするために、行列（４００）に示されたチャネルの順序を、本明細書の残りの行列および式にも使用する。代替実施形態では、チャネルの異なる順序付け、異なる数（たとえば７．１、９．１、２）、および／または構成を有するマルチチャネルオーディオが使用される。

【００７９】

説明される実施形態で、オーディオエンコーダおよびオーディオデコーダは、さまざまな手法を実行する。これらの手法の動作を、提示のために通常は特定のシーケンシャルな順序で説明するが、この説明の形に、特定の順序付けが必要でない場合に、動作の順序の些細な再配置が含まれることを理解されたい。たとえば、順次説明される動作を、いくつかの場合に、再配置するか並列に実行することができる。さらに、説明を単純にするために、流れ図では、通常は、特定の手法を他の手法と共に使用することができるさまざまな形を示さない。

【００８０】

Ｉ．コンピューティング環境

図５に、説明される実施形態を実施することができる適当なコンピューティング環境（５００）の一般化された例を示す。コンピューティング環境（５００）は、本発明の使用または機能性の範囲に関する制限を提案することを意図されたものではない。というのは、本発明を、別個の汎用コンピューティング環境または特殊目的コンピューティング環境で実施することができるからである。

【００８１】

図５を参照すると、コンピューティング環境（５００）に、少なくとも１つの処理ユニット（５１０）とメモリ（５２０）が含まれる。図５では、この最も基本的な構成（５３０）が、破線の中に含まれる。処理ユニット（５１０）は、コンピュータ実行可能命令を実行し、実際のプロセッサまたは仮想プロセッサとすることができる。マルチプロセッシングシステムでは、複数の処理ユニットが、コンピュータ実行可能命令を実行して、処理能力が増やされる。メモリ（５２０）は、揮発性メモリ（たとえば、レジスタ、キャッシュ、ＲＡＭ）、不揮発性メモリ（たとえば、ＲＯＭ、ＥＥＰＲＯＭ、フラッシュメモリなど）、またはこの２つの組合せとすることができる。メモリ（５２０）には、説明される実施形態の１つまたは複数によるオーディオ処理手法を実施するソフトウェア（５８０）が保管される。

【００８２】

コンピューティング環境が、追加の特徴を有することができる。たとえば、コンピューティング環境（５００）に、ストレージ（５４０）、１つまたは複数の入力デバイス（５５０）、１つまたは複数の出力デバイス（５６０）、および１つまたは複数の通信接続（５７０）が含まれる。バス、コントローラ、またはネットワークなどの相互接続機構（図示せず）によって、コンピューティング環境（５００）のコンポーネントが相互接続される。通常、オペレーティングシステムソフトウェア（図示せず）によって、コンピューティング環境（５００）で実行される他のソフトウェアのオペレーティング環境が提供され、コンピューティング環境（５００）のコンポーネントのアクティビティが調整される。

【００８３】

ストレージ（５４０）は、取外し可能または取外し不能とすることができ、ストレージ

（５４０）に、磁気ディスク、磁気テープ、磁気カセット、ＣＤ－ＲＯＭ、ＣＤ－ＲＷ、ＤＶＤ、または、情報を保管するのに使用でき、コンピューティング環境（５００）内でアクセスできる他のメディアが含まれる。ストレージ（５４０）には、説明される実施形態の１つまたは複数によるオーディオ処理手法を実施するソフトウェア（５８０）の命令が保管される。

【００８４】

入力デバイス（５５０）は、キーボード、マウス、ペン、またはトラックボールなどの接触入力デバイス、音声入力デバイス、スキャニングデバイス、ネットワークアダプタ、または、コンピューティング環境（５００）に入力を供給する別のデバイスとすることができる。オーディオに関して、入力デバイス（５５０）を、アナログ形式またはディジタル形式のオーディオ入力を受け入れるサウンドカードまたは類似するデバイス、またはコンピューティング環境にオーディオサンプルを提供するＣＤ－ＲＯＭ／ＤＶＤリーダとすることができる。出力デバイス（５６０）は、ディスプレイ、プリンタ、スピーカ、ＣＤ／ＤＶＤライタ、ネットワークアダプタ、または、コンピューティング環境（５００）から出力を供給する別のデバイスとすることができる。

【００８５】

通信接続（５７０）によって、別のコンピューティングエンティティへの通信メディアを介する通信が可能になる。通信メディアは、コンピュータ実行可能命令、圧縮オーディオ情報、または変調されたデータ信号内の他のデータなどの情報を伝える。変調されたデータ信号とは、情報を信号内でエンコードする形でその特性の１つまたは複数を設定されるか変更された信号である。制限ではなく例として、通信メディアに、電気、光、ＲＦ、赤外線、音響、または他の搬送波を用いて実施される有線または無線の手法が含まれる。

【００８６】

本発明を、コンピュータ可読メディアの全般的な文脈で説明することができる。コンピュータ可読メディアとは、コンピュータ環境内でアクセスできるすべての使用可能なメディアである。制限ではなく例として、コンピューティング環境（５００）に関して、コンピュータ可読メディアに、メモリ（５２０）、ストレージ（５４０）、通信メディア、およびこれらの任意の組合せが含まれる。

【００８７】

本発明を、プログラムモジュールに含まれるものなど、コンピューティング環境内でターゲットの実際のプロセッサまたは仮想プロセッサ上で実行される、コンピュータ実行可能命令の全般的な文脈で説明することができる。一般に、プログラムモジュールには、特定のタスクを実行するか特定の抽象データ型を実施する、ルーチン、プログラム、ライブラリ、オブジェクト、クラス、コンポーネント、データ構造などが含まれる。プログラムモジュールの機能性を、さまざまな実施形態で、望み通りにプログラムモジュールの間で組み合わせるか分割することができる。プログラムモジュールのコンピュータ実行可能命令は、ローカルコンピューティング環境または分散コンピューティング環境内で実行することができる。

【００８８】

提示のために、この詳細な説明では、「決定」、「生成」、「調節」、および「適用」などの単語を使用して、コンピューティング環境でのコンピュータ動作を説明する。これらの単語は、コンピュータによって実行される動作の高水準の抽象化であり、人間によって実行される動作と混同してはならない。これらの単語に対応する実際のコンピュータ動作は、実施形態に応じて変化する。

【００８９】

ＩＩ．一般化されたオーディオエンコーダおよびオーディオデコーダ

図６は、説明される実施形態を実施することができる一般化されたオーディオエンコーダ（６００）のブロック図である。図７は、説明される実施形態を実施することができる一般化されたオーディオデコーダ（７００）のブロック図である。

【００９０】

エンコーダおよびデコーダの中のモジュールの間に示された関係によって、エンコーダとデコーダでの情報の流れが示され、他の関係は、図を単純にするために示されていない。所望の圧縮の実施形態およびタイプに応じて、エンコーダまたはデコーダのモジュールを、追加し、省略し、複数のモジュールに分割し、他のモジュールと組み合わせ、かつ／または類似するモジュールと置換することができる。代替実施形態では、異なるモジュールおよび／または他の構成を有するエンコーダまたはデコーダによって、オーディオデータを処理する。

【００９１】

Ａ．一般化されたオーディオエンコーダ

一般化されたオーディオエンコーダ（６００）には、セレクタ（６０８）、マルチチャネルプリプロセッサ（６１０）、パーティショナ（分配器）（ｐａｒｔｉｔｉｏｎｅｒ）／タイルコンフィギュアラ（ｃｏｎｆｉｇｕｒｅｒ）（６２０）、周波数トランスフォーマ（６３０）、知覚モデラ（６４０）、量子化帯域ウェイタ（６４２）、チャネルウェイタ（６４４）、マルチチャネルトランスフォーマ（６５０）、クォンタイザ（６６０）、エントロピエンコーダ（６７０）、コントローラ（６８０）、ミックスド／ピュアロスレスコーダ（６７２）および関連エントロピエンコーダ（６７４）、およびビットストリームマルチプレクサ［「ＭＵＸ」］（６９０）が含まれる。

【００９２】

エンコーダ（６００）は、あるサンプリング深さとサンプリングレートの入力オーディオサンプル（６０５）の時系列を、パルスコード変調［「ＰＣＭ」］フォーマットで受け取る。説明される実施形態のほとんどについて、入力オーディオサンプル（６０５）は、マルチチャネルオーディオ（たとえば、ステレオ、サラウンド）用であるが、入力オーディオサンプル（６０５）を、その代わりにモノラルとすることができる。エンコーダ（６００）は、オーディオサンプル（６０５）を圧縮し、エンコーダ（６００）のさまざまなモジュールによって作られる情報を多重化して、Ｗｉｎｄｏｗｓ（登録商標）Ｍｅｄｉａ Ａｕｄｉｏ［「WMA」］フォーマットまたはＡｄｖａｎｃｅｄ Ｓｔｒｅａｍｉｎｇ Ｆｏｒｍａｔ［「ＡＳＦ」］などのフォーマットでビットストリーム（６９５）を出力する。その代わりに、エンコーダ（６００）が、他の入力フォーマットおよび／または出力フォーマットを扱うことができる。

【００９３】

セレクタ（６０８）は、オーディオサンプル（６０５）に関する複数のエンコーディングモードの間で選択する。図６では、セレクタ（６０８）が、ミックスド／ピュアロスレスコーディングモードとロッシイコーディングモードの間で切り替える。ロスレスコーディングモードには、ミックスド／ピュアロスレスコーダ（６７２）が含まれ、ロスレスコーディングモードは、通常は、高品質（および高ビットレート）の圧縮に使用される。ロッシイコーディングモードには、ウェイタ（６４２）およびクォンタイザ（６６０）などのコンポーネントが含まれ、ロッシイコーディングモードは、通常は調整可能な品質（および制御されたビットレート）の圧縮に使用される。セレクタ（６０８）での選択判断は、ユーザ入力または他の判断基準に依存する。ある情況（たとえば、ロッシイ圧縮で適当な品質を配信できないか、ビットが過剰に作られる時）では、エンコーダ（６００）が、あるフレームまたはフレームの組について、ロッシイコーディングからミックスド／ピュアロスレスコーディングに切り替えることができる。

【００９４】

マルチチャネルオーディオデータのロッシイコーディングについて、マルチチャネルプリプロセッサ（６１０）は、任意選択として、時間領域オーディオサンプル（６０５）を再行列化する。いくつかの実施形態で、マルチチャネルプリプロセッサ（６１０）は、オーディオサンプル（６０５）を選択的に再行列化して、１つまたは複数のコーディングされたチャネルを捨てるか、エンコーダ（６００）内の相互チャネル相関性を増やすが、それでもデコーダ（７００）での再構成（ある形での）を可能にする。これによって、エンコーダに、チャネルレベルでの品質に対する追加の制御が与えられる。マルチチャネルプ

リプロセッサ（６１０）は、マルチチャネル後処理の命令などのサイド情報を、ＭＵＸ（６９０）に送ることができる。いくつかの実施形態でのマルチチャネルプリプロセッサの動作に関する追加の詳細については、「マルチチャネル前処理」という題名のセクションを参照されたい。代替案では、エンコーダ（６００）が、別の形のマルチチャネル前処理を実行する。

【００９５】

パーティショナ／タイルコンフィギュアラ（６２０）は、オーディオ入力サンプル（６０５）のフレームを、時間依存性サイズ関数およびウィンドウ整形関数（ｔｉｍｅ－ｖａｒｙｉｎｇ　ｓｉｚｅ　ａｎｄ　ｗｉｎｄｏｗ　ｓｈａｐｉｎｇ　ｆｕｎｃｔｉｏｎｓ）を有するサブフレームブロック（すなわちウィンドウ）に区分する。サブフレームブロックのサイズおよびウィンドウは、フレーム内の推移信号の検出、コーディングモード、ならびに他の要因に依存する。

【００９６】

エンコーダ（６００）が、ロッシイコーディングからミックスド／ピュアロスレスコーディングに切り替える場合に、サブフレームブロックは、理論的にはオーバーラップする必要も、ウィンドウウィング関数を有する必要もない（すなわち、オーバーラップしない長方形のウィンドウブロック）が、ロッシイコーディングされたフレームと他のフレームの間の推移は、特別な扱いを必要とする可能性がある。パーティショナ／タイルコンフィギュアラ（６２０）は、区分されたデータのブロックを、ミックスド／ピュアロスレスコーダ（６７２）に出力し、ブロックサイズなどのサイド情報をＭＵＸ（６９０）に出力する。ミックスドまたはピュアのロスレスコーディングされたフレームの区分およびウィンドウイングに関する追加の詳細については、関連特許出願の発明の名称“Unified Lossy and Lossless Audio Compression”の米国特許出願第６０／４０８４３２号を参照されたい。

【００９７】

エンコーダ（６００）が、ロッシイコーディングを使用する時には、可変サイズウィンドウによって、可変時間分解能が可能になる。小さいブロックを用いると、短いがアクティブな推移セグメントで、時間詳細のより多くの保存が可能になる。大きいブロックは、よりよい周波数分解能とより悪い時間分解能を有し、通常は、大きいブロックによって、より長くより少数のアクティブセグメントでのより高い圧縮効率が可能になる。これは、部分的にはフレームヘッダおよびサイド情報が、小さいブロックよりもサイズに比例して少なくなるからであり、部分的にはこれによってよりよい冗長性削減が可能になるからである。ブロックをオーバーラップさせて、そうでなければ後の量子化によって導入される可能性があるブロック間の知覚可能な不連続性を減らすことができる。パーティショナ／タイルコンフィギュアラ（６２０）は、区分されたデータのブロックを周波数トランスフォーマ（６３０）に出力し、ブロックサイズなどのサイド情報をＭＵＸ（６９０）に出力する。いくつかの実施形態での推移検出および区分判断基準に関する追加情報については、参照によって本明細書に組み込まれる関連特許出願の発明の名称"Adaptive Window-Size Selection in Transform Coding," の米国特許出願第１０／０１６，９１８（２００１年１２月１４日出願）を参照されたい。代替案では、パーティショナ／タイルコンフィギュアラ（６２０）で、フレームをウィンドウに区分する時に、他の区分判断基準またはブロックサイズを使用する。

【００９８】

いくつかの実施形態で、パーティショナ／タイルコンフィギュアラ（６２０）は、マルチチャネルオーディオのフレームをチャネルごとに区分する。パーティショナ／タイルコンフィギュアラ（６２０）は、品質／ビットレートから許容される場合に、フレーム内の各チャネルを独立に区分する。これによって、たとえば、パーティショナ／タイルコンフィギュアラ（６２０）が、より小さいウィンドウを用いて特定のチャネルに現れる推移を分離するが、他のチャネルで周波数分解能または圧縮効率のためにより大きいウィンドウを使用することが可能になる。これによって、チャネルごとに推移を分離することによっ

て圧縮効率を改善することができるが、多くの場合に、個々のチャネル内の区分を指定する追加情報が、必要になる。時間的に同一位置にある同一サイズのウィンドウは、マルチチャネル変換を介するさらなる冗長性削減の資格を有する場合がある。したがって、パーティショナ／タイルコンフィギュアラ（６２０）は、時間的に同一位置にある同一サイズのウィンドウを、タイルとしてグループ化する。いくつかの実施形態でのタイリングに関する追加の詳細については、「タイル構成」という題名のセクションを参照されたい。

【００９９】

周波数トランスフォーマ（６３０）は、オーディオサンプルを受け取り、周波数領域のデータに変換する。周波数トランスフォーマ（６３０）は、周波数係数データのブロックをウェイタ（６４２）に出力し、ブロックサイズなどのサイド情報をＭＵＸ（６９０）に出力する。周波数トランスフォーマ（６３０）は、周波数係数とサイド情報の両方を知覚モデラ（６４０）に出力する。いくつかの実施形態で、周波数トランスフォーマ（６３０）は、時間に伴って変化する変調ラップド変換（Ｍｏｄｕｌａｔｅｄ　Ｌａｐｐｅｄ　Ｔｒａｎｓｆｏｒｍ）［「ＭＬＴ」］をサブフレームブロックに適用するが、このＭＬＴは、サブフレームブロックの正弦ウィンドウ関数によって変調されたＤＣＴに似た演算である。代替実施形態では、ＭＬＴの他の変形形態またはＤＣＴあるいは、変調ありまたはなしの、オーバーラップありまたはなしの、他のタイプの周波数変換を使用するか、サブバンドコーディングまたはウェーブレットコーディングを使用する。

【０１００】

知覚モデラ（６４０）によって、人間の聴覚系のプロパティをモデル化して、所与のビットレートの再構成されたオーディオ信号の知覚される品質を改善する。一般に、知覚モデラ（６４０）は、聴覚モデルに従ってオーディオデータを処理し、その情報をウェイタ（６４２）に供給し、このウェイタ（６４２）は、オーディオデータの重みづけ係数を生成するのに使用することができる。知覚モデラ（６４０）は、さまざまな聴覚モデルのいずれかを使用し、励起パターン情報または他の情報をウェイタ（６４２）に渡す。

【０１０１】

量子化帯域ウェイタ（６４２）は、知覚モデラ（６４０）から受け取った情報に基づいて量子化行列の重みづけ係数を生成し、その重みづけ係数を、周波数トランスフォーマ（６３０）から受け取ったデータに適用する。量子化行列の重みづけ係数には、オーディオデータの複数の量子化帯域のそれぞれの重みが含まれる。量子化帯域は、数またはエンコーダ（６００）の他所で使用される臨界帯域からの位置において、同一または異なるものとすることができ、重みづけ係数を、ブロックごとに、振幅および量子化帯域の数において変更することができる。量子化帯域ウェイタ（６４２）は、係数データの重みづけされたブロックをチャネルウェイタ（６４４）に出力し、重みづけされた係数の組などのサイド情報をＭＵＸ（６９０）に出力する。重みづけされた係数の組を、さらに効率的な表現のために圧縮することができる。重みづけ係数が、ロッシイ圧縮される場合には、再構成される重みづけ係数が、通常は、係数データのブロックに重みをつけるのに使用される。いくつかの実施形態での重みづけ係数の計算および圧縮に関する追加の詳細については、「量子化および重みづけ」という題名のセクションを参照されたい。代替案では、エンコーダ（６００）が、別の形の重みづけを使用するか、重みづけをスキップする。

【０１０２】

チャネルウェイタ（６４４）は、知覚モデラ（６４０）から受け取った情報およびローカルに再構成された信号の品質に基づいて、チャネルのチャネル固有重みづけ係数（スカラである）を生成する。スカラ重み（量子化ステップ変更子とも称する）を用いると、エンコーダ（６００）が、再構成されるチャネルに、近似的に均一の品質を与えられるようになる。チャネル重み係数は、チャネルごとおよびブロックごとに、またはある他のレベルで、振幅を変えることができる。チャネルウェイタ（６４４）は、係数データの重みづけされたブロックをマルチチャネルトランスフォーマ（６５０）に出力し、チャネル重み係数の組などのサイド情報をＭＵＸ（６９０）に出力する。流れ図のチャネルウェイタ（６４４）および量子化帯域ウェイタ（６４２）は、入れ替えるか、一緒に組み合わせるこ

とができる。いくつかの実施形態での重みづけ係数の計算および圧縮に関する追加の詳細については、「量子化および重みづけ」という題名のセクションを参照されたい。代替案では、エンコーダ（６００）が、別の形の重みづけを使用するか、重みづけをスキップする。

【０１０３】

マルチチャネルオーディオデータに関して、チャネルウェイタ（６４４）によって作られる雑音形の（ｎｏｉｓｅ－ｓｈａｐｅｄ）周波数係数データの複数のチャネルが、しばしば相関し、したがって、マルチチャネルトランスフォーマ（６５０）が、マルチチャネル変換を適用することができる。たとえば、マルチチャネルトランスフォーマ（６５０）は、タイルのチャネルおよび／または量子化帯域のすべてではなく一部に、マルチチャネル変換を選択的に柔軟に適用する。これによって、マルチチャネルトランスフォーマ（６５０）に、タイルの比較的相関する部分への変換の適用に対する正確な制御が与えられる。計算的な複雑さを減らすために、マルチチャネルトランスフォーマ（６５０）は、１レベル変換ではなく階層変換を使用することができる。変換行列に関連するビットレートを減らすために、マルチチャネルトランスフォーマ（６５０）は、事前定義の行列（たとえば、恒等変換／無変換、アダマール、ＤＣＴタイプＩＩ）またはカスタム行列を選択的に使用し、カスタム行列に効率的な圧縮を適用する。最後に、マルチチャネル変換は、ウェイタ（６４２）の下流なので、デコーダ（７００）での逆マルチチャネル変換の後のチャネル間で漏れる雑音を知覚できること（たとえば、後続の量子化に起因する）が、逆重みづけによって制御される。いくつかの実施形態でのマルチチャネル変換に関する追加の詳細については、「柔軟なマルチチャネル変換」という題名のセクションを参照されたい。代替案では、エンコーダ（６００）が、他の形のマルチチャネル変換を使用するか、まったく変換を行わない。マルチチャネルトランスフォーマ（６５０）は、ＭＵＸ（６９０）へのサイド情報を作って、たとえば、使用されたマルチチャネル変換およびタイルのマルチチャネル変換された部分を示す。

【０１０４】

クォンタイザ（６６０）は、マルチチャネルトランスフォーマ（６５０）の出力を量子化し、エントロピエンコーダ（６７０）への量子化された係数データおよびＭＵＸ（６９０）への量子化ステップサイズを含むサイド情報を作る。図６では、クォンタイザ（６６０）が、タイルごとに量子化係数を計算する適応式均一スカラクォンタイザである。タイル量子化係数を、量子化ループの反復ごとに変更して、エントロピエンコーダ（６７０）出力のビットレートに影響を及ぼすことができ、チャネルごとの量子化ステップ変更子を使用して、チャネルの間の再構成品質のバランスをとることができる。いくつかの実施形態での量子化に関する追加の詳細については、「量子化および重みづけ」という題名のセクションを参照されたい。代替実施形態では、クォンタイザが、不均一クォンタイザ、ベクトルクォンタイザ、および／または非適応クォンタイザであるか、異なる形の適応均一スカラ量子化を使用する。他の代替実施形態では、クォンタイザ（６６０）、量子化帯域ウェイタ（６４２）、チャネルウェイタ（６４４）、およびマルチチャネルトランスフォーマ（６５０）が、融合され、融合されたモジュールが、さまざまな重みをすべて一緒に判定する。

【０１０５】

エントロピエンコーダ（６７０）は、クォンタイザ（６６０）から受け取った量子化された係数データをロスレス圧縮する。いくつかの実施形態で、エントロピエンコーダ（６７０）は、関連特許出願の発明の名称"Entropy Coding by Adapting Coding Between Level and Run Length/Level Modes"の米国特許出願第６０／４０８，５３８号に記載の適応エントロピコーディングを使用する。代替案では、エントロピエンコーダ（６７０）が、マルチレベルランレングスコーディング、可変長対可変長コーディング、ランレングスコーディング、ハフマンコーディング、辞書コーディング、算術コーディング、ＬＺコーディング、または他のエントロピコーディング手法の他の形または組合せを使用する。エントロピエンコーダ（６７０）は、オーディオ情報のエンコーディングに費やされるビット

数を計算し、この情報をレート／品質コントローラ（６８０）に渡すことができる。

【０１０６】

コントローラ（６８０）は、クォンタイザ（６６０）と共に働いて、エンコーダ（６００）の出力のビットレートおよび／または品質を調整する。コントローラ（６８０）は、エンコーダ（６００）の他のモジュールから情報を受け取り、受け取った情報を処理して、現在の条件に対して所望の量子化係数を判定する。コントローラ（６８０）は、品質制約および／またはビットレート制約を満足するという目標をもって、クォンタイザ（６６０）に量子化係数を出力する。

【０１０７】

ミックスド／ピュアロスレスコーダ（６７２）および関連エントロピエンコーダ（６７４）は、ミックスド／ピュアロスレスコーディングモードでオーディオデータを圧縮する。エンコーダ（６００）は、シーケンス全体にミックスド／ピュアロスレスコーディングモードを使用するか、フレームごと、ブロックごと、タイルごと、または他の基準でコーディングモードを切り替える。ミックスド／ピュアロスレスコーディングモードに関する追加の詳細については、関連特許出願の発明の名称“Unified Lossy and Lossless Audio Compression”の米国特許出願第６０／４０８４３２号を参照されたい。代替案では、エンコーダ（６００）が、ミックスドおよび／またはピュアのロスレスエンコーディングの他の手法を使用する。

【０１０８】

ＭＵＸ（６９０）は、オーディオエンコーダ（６００）の他のモジュールから受け取ったサイド情報を、エントロピエンコーダ（６７０、６７４）から受け取ったエントロピエンコーディングされたデータと多重化する。ＭＵＸ（６９０）は、ＷＭＡフォーマットまたはオーディオデコーダが認識する別のフォーマットで情報を出力する。ＭＵＸ（６９０）には、エンコーダ（６００）によって出力されるビットストリーム（６９５）を保管する仮想バッファが含まれる。仮想バッファは、比較的一定のビットレートでデータを出力し、品質は、入力の複雑さの変化に起因して変化する可能性がある。バッファの現在の満杯度および他の特性を、コントローラ（６８０）によって使用して、品質および／またはビットレートを調整することができる。代替案では、出力ビットレートが、経時的に変化することができ、品質が、比較的一定に保たれる。あるいは、出力ビットレートが、特定のビットレート未満に制限されるだけであり、このビットレートは、一定にまたは時間的に変換するのいずれかである。

【０１０９】

Ｂ．一般化されたオーディオデコーダ

図７を参照すると、一般化されたオーディオデコーダ（７００）に、ビットストリームデマルチプレクサ［「ＤＥＭＵＸ」］（７１０）、１つまたは複数のエントロピデコーダ（７２０）、ミックスド／ピュアロスレスデコーダ（７２２）、タイル構成デコーダ（７３０）、逆マルチチャネルトランスフォーマ（７４０）、逆クォンタイザ／ウェイタ（７５０）、逆周波数トランスフォーマ（７６０）、オーバーラッパ／アダー（７７０）、およびマルチチャネルポストプロセッサ（７８０）が含まれる。デコーダ（７００）にはレート／品質制御または知覚モデリングのモジュールが含まれないので、デコーダ（７００）は、エンコーダ（６００）より多少単純である。

【０１１０】

デコーダ（７００）は、ＷＭＡフォーマットまたは別のフォーマットの圧縮オーディオ情報のビットストリーム（７０５）を受け取る。ビットストリーム（７０５）には、エントロピエンコーディングされたデータならびにサイド情報が含まれ、デコーダ（７００）は、それらからオーディオサンプル（７９５）を再構成する。

【０１１１】

ＤＥＭＵＸ（７１０）は、ビットストリーム（７０５）の情報を解析し、情報をデコーダ（７００）のモジュールに送る。ＤＥＭＵＸ（７１０）には、オーディオの複雑さの変動、ネットワークジッタ、および／または他の要因に起因するビットレートの短期間変動

を補償するために、１つまたは複数のバッファが含まれる。

【０１１２】

１つまたは複数のエントロピデコーダ（７２０）は、ＤＥＭＵＸ（７１０）から受け取るエントロピコードをロスレス圧縮解除する。エントロピデコーダ（７２０）は、通常は、エンコーダ（６００）で使用されるエントロピエンコード手法の逆を適用する。説明を単純にするために、１つのエントロピデコーダモジュールを図７に示したが、異なるエントロピデコーダを、ロッシイコーディングモードとロスレスコーディングモードに使用することができ、１つのモードの中で異なるエントロピデコーダを使用することもできる。また、説明を単純にするために、図７には、モード選択論理が示されていない。ロッシイコーディングモードで圧縮されたデータをデコードする時に、エントロピデコーダ（７２０）は、量子化された周波数係数データを作る。

【０１１３】

ミックスド／ピュアロスレスデコーダ（７２２）および関連するエントロピデコーダ（７２０）は、ミックスド／ピュアロスレスコーディングモードのロスレスエンコーディングされたオーディオデータを圧縮解除する。ミックスド／ピュアロスレスデコーディングモードの圧縮解除に関する追加の詳細については、関連特許出願の発明の名称“Unified Lossy and Lossless Audio Compression”の米国特許出願第６０／４０８４３２号を参照されたい。代替案では、デコーダ（７００）が、ミックスドおよび／またはピュアのロスレスデコーディングの他の手法を使用する。

【０１１４】

タイル構成デコーダ（７３０）は、ＤＥＭＵＸ（７１０）から、フレームのタイルのパターンを示す情報を受け取り、必要な場合にデコードする。タイルパターン情報は、エントロピエンコーディングされるか、他の形でパラメータ化される可能性がある。タイル構成デコーダ（７３０）は、タイルパターン情報を、デコーダ（７００）のさまざまな他のモジュールに渡す。いくつかの実施形態でのタイル構成デコーディングに関する追加の詳細については、「タイル構成」という題名のセクションを参照されたい。代替案では、デコーダ（７００）が、フレーム内のウィンドウパターンをパラメータ化する他の手法を使用する。

【０１１５】

逆マルチチャネルトランスフォーマ（７４０）は、エントロピデコーダ（７２０）からの量子化された周波数係数データならびにタイル構成デコーダ（７３０）からのタイルパターン情報および、たとえば使用されたマルチチャネル変換およびタイルの変換された部分を示す、ＤＥＭＵＸ（７１０）からのサイド情報を受け取る。この情報を使用して、逆マルチチャネルトランスフォーマ（７４０）は、必要に応じて変換行列を圧縮解除し、１つまたは複数の逆マルチチャネル変換をオーディオデータに選択的に柔軟に適用する。逆クォンタイザ／ウェイタ（７５０）に間する逆マルチチャネルトランスフォーマ（７４０）の配置は、チャネルにまたがって漏れる可能性がある量子化雑音を整形するのに役立つ。いくつかの実施形態の逆マルチチャネルトランスフォーマに関する追加の詳細については、「柔軟なマルチチャネル変換」という題名のセクションを参照されたい。

【０１１６】

逆クォンタイザ／ウェイタ（７５０）は、タイルおよびチャネルの量子化係数ならびに量子化行列をＤＥＭＵＸ（７１０）から受け取り、量子化された周波数係数データを逆マルチチャネルトランスフォーマ（７４０）から受け取る。逆クォンタイザ／ウェイタ（７５０）は、受け取った量子化係数／行列情報を必要に応じて圧縮解除し、逆量子化および重みづけを実行する。いくつかの実施形態での逆量子化および重みづけの追加の詳細については、「量子化および重みづけ」という題名のセクションを参照されたい。代替実施形態では、逆クォンタイザ／ウェイタによって、エンコーダで使用される他の量子化手法の逆が適用される。

【０１１７】

逆周波数トランスフォーマ（７６０）は、逆クォンタイザ／ウェイタ（７５０）によっ

て出力される周波数係数データならびにＤＥＭＵＸ（７１０）からのサイド情報およびタイル構成デコーダ（７３０）からのタイルパターン情報を受け取る。逆周波数トランスフォーマ（７６０）は、エンコーダで使用される周波数変換の逆を適用し、ブロックをオーバーラッパ／アダー（７７０）に出力する。

【０１１８】

タイル構成デコーダ（７３０）からタイルパターン情報を受け取るほかに、オーバーラッパ／アダー（７７０）は、逆周波数トランスフォーマ（７６０）および／またはミックスド／ピュアロスレスデコーダ（７２２）からデコードされた情報も受け取る。オーバーラッパ／アダー（７７０）は、必要に応じてオーディオデータをオーバーラップさせ、加算し、異なるモードでエンコードされたオーディオデータのフレームまたは他のシーケンスをインターリーブする。ミックスドまたはピュアのロスレスコーディングされたフレームのオーバーラップ、加算、およびインターリーブに関する追加の詳細は、関連特許出願の発明の名称“Unified Lossy and Lossless Audio Compression”の米国特許出願第６０／４０８４３２号を参照されたい。代替案では、デコーダ（７００）が、フレームのオーバーラップ、加算、およびインターリーブに他の手法を使用する。

【０１１９】

マルチチャネルポストプロセッサ（７８０）は、任意選択として、オーバーラッパ／アダー（７７０）によって出力される時間領域オーディオサンプルを再行列化する。マルチチャネルポストプロセッサは、オーディオデータを選択的に再行列化して、再生用のファントムチャネルを作成し、スピーカの間でのチャネルの空間的回転、より少数のスピーカでの再生または他の目的のためのチャネルの折り曲げなどの特殊効果を実行する。ビットストリーム制御された後処理について、後処理変換行列は、経時的に変化し、シグナリングされるかビットストリーム（７０５）に含まれる。いくつかの実施形態でのマルチチャネルポストプロセッサの動作に関する追加の詳細は、「マルチチャネル後処理」という題名のセクションを参照されたい。代替案では、デコーダ（７００）が、別の形のマルチチャネル後処理を実行する。

【０１２０】

ＩＩＩ．マルチチャネル前処理

いくつかの実施形態で、図６のエンコーダ（６００）などのエンコーダが、時間領域の入力オーディオサンプルに対してマルチチャネル前処理を実行する。

【０１２１】

一般に、入力としてＮ個のソースオーディオチャネルがある時に、エンコーダによって作られるコーディングされたチャネルの数もＮになる。コーディングされたチャネルが、ソースチャネルと１対１対応する場合があり、あるいは、コーディングされたチャネルが、マルチチャネル変換コーディングされたチャネルである場合がある。しかし、ソースのコーディングの複雑さによって圧縮が困難になる時、またはエンコーダバッファが満杯である時には、エンコーダが、元の入力オーディオチャネルの１つまたは複数を変更するか捨てる（すなわちコーディングしない）場合がある。これは、コーディングの複雑さを減らし、オーディオの全体的な知覚される品質を改善するために行うことができる。品質駆動の前処理について、エンコーダは、測定されたオーディオ品質に反応してマルチチャネル前処理を実行して、全体的なオーディオ品質およびチャネルセパレーションを滑らかに制御する。

【０１２２】

たとえば、エンコーダは、マルチチャネルオーディオイメージを変更して、１つまたは複数のチャネルをよりクリティカルでないようにすることができ、その結果、チャネルがエンコーダで捨てられるが、デコーダで「ファントムチャネル」として再構成されるようになる。チャネルの徹底的な削除は、品質に劇的に影響する可能性があり、したがって、これは、コーディングの複雑さが非常に高いか、バッファが非常に満杯であり、他の手段を介して良い品質の再生を達成できない時に限って行われる。

【０１２３】

エンコーダは、コーディングされるチャネルの数が、出力のチャネル数より少ない時にどの処置を講ずるかをデコーダに示すことができる。その後、マルチチャネル後処理変換をデコーダで使用して、下の「マルチチャネル後処理」という題名のセクションで説明するように、ファントムチャネルを作成することができる。あるいは、エンコーダが、別の目的のマルチチャネル後処理を実行するようにデコーダに知らせることができる。

【０１２４】

図８に、マルチチャネル前処理の一般化された手法（８００）を示す。エンコーダが、時間領域マルチチャネルオーディオデータ（８０５）に対するマルチチャネル前処理を実行し（８１０）、時間領域の変換されたオーディオデータ（８１５）を作る。たとえば、前処理に、一般的なＮ対Ｎ変換が含まれ、このＮは、チャネルの数である。エンコーダは、Ｎ個のサンプルに行列Ａをかける。

$$y_{pre} = A_{pre} x_{pre} \qquad (4)$$

ここで、x_{pre}およびy_{pre}は、前処理に入力されるＮ個の入力および前処理から出力されるＮ個の出力であり、A_{pre}は、実数（すなわち連続的な）値の要素を有する一般的なＮ×Ｎ変換行列である。行列A_{pre}は、x_{pre}と比較してy_{pre}の相互チャネル相関性を人工的に増やすように選択することができる。これによって、エンコーダの残りに関する複雑さが減るが、チャネルセパレーションの低下が犠牲になる。

【０１２５】

出力y_{pre}が、エンコーダの残りに供給され、これによって、図６に示された手法または他の圧縮手法を使用してデータがエンコードされ（８２０）、エンコードされたマルチチャネルオーディオデータ（８２５）が作られる。

【０１２６】

エンコーダおよびデコーダによって使用される構文（ｓｙｎｔａｘ）によって、一般的なまたは事前定義の後処理マルチチャネル変換行列の記述が可能になり、この後処理マルチチャネル変換行列は、フレームごとに変更するか、オン／オフにすることができる。エンコーダは、この柔軟性を使用して、ステレオ／サラウンドイメージ減損を制限し、相互チャネル相関性を人工的に増やすことによって、ある情況でチャネルセパレーションとよりよい総合的な品質をトレードオフする。代替案では、デコーダおよびエンコーダが、マルチチャネル前処理およびマルチチャネル後処理の別の構文、たとえば、フレームごと以外の基礎での変換行列の変更を可能にする構文を使用する。

【０１２７】

図９ａから図９ｅに、ある情況の下でエンコーダで相互チャネル相関性を人工的に増やすのに使用されるマルチチャネル前処理変換行列（９００から９０４）を示す。エンコーダは、前処理行列の間で切り替えて、５．１チャネル再生環境で、左チャネル、右チャネル、および中央チャネルの間、および左後ろチャネルと右後ろチャネルの間で相互チャネル相関性を人工的にどれほど増やすかを変更する。

【０１２８】

一実施形態で、低ビットレートで、エンコーダが、ある時間の期間にわたって再構成されたオーディオの品質を評価し、その結果に応じて、前処理行列の１つを選択する。エンコーダによって評価される品質測定は、雑音興奮比率［「ＮＥＲ」］であり、これは、元のディジタルオーディオクリップのエネルギに対する再構成されたオーディオクリップの雑音パターンのエネルギの比である。低いＮＥＲ値は、良い品質を示し、高いＮＥＲ値は、低い品質を示す。エンコーダは、１つまたは複数の前にエンコードされたフレームのＮＥＲを評価する。ＮＥＲおよび他の品質測定に関する追加情報については、参照によって本明細書に組み込まれる関連特許出願の発明の名称"Techniques for Measurement of Perceptual Audio Quality," の米国特許出願第１０／０１７，８６１号（２００１年１２月１４日出願）を参照されたい。代替案では、エンコーダが、別の品質測定、バッファ満杯度、および／またはある他の判断基準を使用して、前処理変換行列を選択し、あるいは、エンコーダが、マルチチャネルオーディオの異なる期間を評価する。

【０１２９】

図９ａから図９ｅに示された例に戻ると、低ビットレートで、エンコーダは、オーディオクリップの特定の範囲のＮＥＲｎに基づいて、前処理変換行列をゆっくり変更する。エンコーダは、ｎの値を閾値n_{low}およびn_{high}と比較するが、これらの閾値は、実装依存である。一実施形態では、n_{low}およびn_{high}が、所定の値n_{low}＝０．０５およびn_{high}＝０．１を有する。代替案では、n_{low}およびn_{high}が、ビットレートまたは他の判断基準に反応して経時的に変化する１つまたは複数の異なる値を有し、あるいは、エンコーダが、異なる数の行列の間で切り替える。

【０１３０】

低い値のｎ（たとえば、$n \leqq n_{low}$）は、良い品質のコーディングを示す。したがって、エンコーダは、図９ａに示された単位行列A_{low}（９００）を使用し、効果的に前処理をオフにする。

【０１３１】

その一方で、高い値のｎ（たとえば、$n \geqq n_{high}$）は、低い品質のコーディングを示す。したがって、エンコーダは、図９ｃに示された行列$A_{high,1}$（９０２）を使用する。行列$A_{high,1}$（９０２）によって、激しいサラウンドイメージひずみが導入されるが、それと同時に、左チャネル、右チャネル、および中央チャネルの間の非常に高い相関が押し付けられ、これによって、複雑さを減らすことによって後続のコーディング効率が改善される。マルチチャネル変換された中央チャネルは、元の左チャネル、右チャネル、および中央チャネルの平均である。行列$A_{high,1}$（９０２）によって、後チャネルの間のチャネルセパレーションも妥協して処理され、入力の左後ろチャネルと右後ろチャネルの平均がとられる。

【０１３２】

中間の値のｎ（たとえば、$n_{low} < n < n_{high}$）は、中間の品質のコーディングを示す。したがって、エンコーダは、図９ｂに示された中間行列$A_{inter,1}$（９０１）を使用することができる。中間行列$A_{inter,1}$（９０１）では、係数αによって、n_{low}とn_{high}の間のｎの相対位置が測定される。

【０１３３】

【数３】

$$\alpha = \frac{n - n_{low}}{n_{high} - n_{low}} \qquad (5)$$

【０１３４】

中間行列$A_{inter,1}$（９０１）は、単位行列A_{low}（９００）から低品質行列$A_{high,1}$（９０２）へ徐々に推移する。

【０１３５】

図９ｂおよび図９ｃに示された行列$A_{inter,1}$（９０１）および$A_{high,1}$（９０２）について、エンコーダは、後に、エンコーダが相互チャネル相関性を人工的に増やしたチャネルの間の冗長性を活用し、エンコーダは、これらのチャネルに関してマルチチャネル後処理を実行するようにエンコーダに指示する必要がない。

【０１３６】

デコーダが、マルチチャネル後処理を実行する能力を有する時には、エンコーダは、中央チャネルの再構成をデコーダに委任することができる。そうである場合に、ＮＥＲ値ｎによって、低い品質のコーディングが示される時に、エンコーダは、図９ｅに示された行列$A_{high,2}$（９０４）を使用するが、この行列を用いると、入力中央チャネルが左右のチャネルに漏れる。出力では、中央チャネルが０であり、コーディングの複雑さが減る。

【０１３７】

【数４】

$$\begin{bmatrix} \left(\frac{a}{1.5}+\frac{.5\cdot c}{1.5}\right) \\ \left(\frac{b}{1.5}+\frac{.5\cdot c}{1.5}\right) \\ 0 \\ d \\ \frac{e+f}{2} \\ \frac{e+f}{2} \end{bmatrix} = A_{high,2} \cdot \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix}$$

【０１３８】

エンコーダは、前処理変換行列Ａｈｉｇｈ，２（９０４）を使用する時に、デコードされた左右のチャネルの平均をとることによってファントム中央を作成するようにデコーダに（ビットストリームを介して）指示する。エンコーダでの後のマルチチャネル変換では、平均をとられた後ろの左右のチャネル（後処理なし）の間の冗長性を活用することができ、あるいは、エンコーダが、後ろの左右のチャネルに関するあるマルチチャネル後処理を実行するようにデコーダに指示することができる。

【０１３９】

ＮＥＲ値ｎによって、中間の品質のコーディングが示される時には、エンコーダは、図９ｄに示された中間行列Ａｉｎｔｅｒ，２（９０３）を使用して、図９ａおよび９ｅに示された行列の間で推移することができる。

【０１４０】

図１０に、フレームごとに変換行列が潜在的に変化するマルチチャネル前処理の手法（１０００）を示す。変換行列の変更は、注意深く処理されない場合に、最終出力の可聴雑音（たとえばポンという音）につながる可能性がある。ポンという雑音を導入しないようにするために、エンコーダは、ある変換行列から別の変換行列へ、フレームの間に徐々に推移する。

【０１４１】

エンコーダは、まず、上で説明した前処理変換行列をセットする（１０１０）。次に、エンコーダは、現在のフレームの行列が、前のフレーム（前のフレームがある場合に）の行列と異なるかどうかを判定する（１０２０）。現在の行列が同一であるか、前の行列がない場合には、エンコーダは、現在のフレームの入力オーディオサンプルに行列を適用する（１０３０）。そうでない場合には、エンコーダは、現在のフレームの入力オーディオサンプルにブレンドされた変換行列を適用する（１０４０）。ブレンディング関数は、実施形態に依存する。一実施形態では、現在のフレームのサンプルｉで、エンコーダが、短期間ブレンドされた行列Ａｐｒｅ，ｉを使用する。

【０１４２】

【数５】

$$A_{pre,i} = \frac{NumSamples - i}{NumSamples} A_{pre,prev} + \frac{i}{NumSamples} A_{pre,current} \qquad (6)$$

【０１４３】

ここで、Ａｐｒｅ，ｐｒｅｖおよびＡｐｒｅ，ｃｕｒｒｅｎｔは、それぞれ前のフレームおよび現在のフレームの前処理行列であり、ＮｕｍＳａｍｐｌｅｓは、現在のフレームのサンプル数である。代替案では、エンコーダが、別のブレンディング関数を使用して、前処理変換行列の不連続性を平滑化する。

【０１４４】

次に、エンコーダは、図６に示した手法または他の圧縮手法を使用して、フレームのマ

ルチチャネルオーディオデータをエンコードする（１０５０）。エンコーダは、フレームごとに手法（１０００）を繰り返す。代替案では、エンコーダが、他の基礎に基づいてマルチチャネル前処理を変更する。

【０１４５】

ＩＶ．タイル構成

いくつかの実施形態で、図６のエンコーダ（６００）などのエンコーダが、マルチチャネルオーディオのウィンドウを、後続のエンコーディングのためにタイルにグループ化する。これによって、フレームのチャネルのさまざまな組合せに対するマルチチャネル変換を可能にしながら、エンコーダに、フレームの異なるチャネルについて異なるウィンドウ構成を使用する柔軟性が与えられる。図７のデコーダ（７００）などのデコーダが、デコード中にタイルを処理する。

【０１４６】

各チャネルが、他のチャネルと独立のウィンドウ構成を有することができる。同一の開始時刻および停止時刻を有するウィンドウは、タイルの一部とみなされる。タイルは、１つまたは複数のチャネルを有することができ、エンコーダは、タイル内のチャネルに関してマルチチャネル変換を実行する。

【０１４７】

図１１ａに、ステレオオーディオのフレームの例のタイル構成（１１００）を示す。図１１ａでは、各タイルに単一のウィンドウが含まれる。ステレオオーディオのどちらのチャネルのウィンドウも、他のチャネルのウィンドウと同一の時刻に始まらず、停止しない。

【０１４８】

図１１ｂに、５．１チャネルオーディオのフレームの例のタイル構成（１１０１）を示す。タイル構成（１１０１）には、０から６までの番号をつけられた７つのタイルが含まれる。タイル０には、チャネル０、２、３、および４からのサンプルが含まれ、タイル０は、フレームの最初の１／４にまたがる。タイル１には、チャネル１からのサンプルが含まれ、タイル１は、フレームの最初の１／２にまたがる。タイル２には、チャネル５からのサンプルが含まれ、タイル２は、フレーム全体にまたがる。タイル３は、タイル０に似ているが、フレームの第２の１／４にまたがる。タイル４および６には、チャネル０、２、および３のサンプルが含まれ、タイル４および６は、それぞれ、フレームの３番目の１／４および４番目の１／４にまたがる。最後に、タイル５には、チャネル１および４からのサンプルが含まれ、タイル５は、フレームの後半分にまたがる。図１１ｂからわかるように、特定のタイルに、不連続なチャネルのウィンドウを含めることができる。

【０１４９】

図１２に、マルチチャネルオーディオのフレームのタイルを構成する一般化された手法（１２００）を示す。エンコーダは、フレーム内のチャネルに関してウィンドウ構成をセットし（１２１０）、各チャネルを可変サイズウィンドウに区分して、時間分解能と周波数分解能をトレードオフする。たとえば、エンコーダのパーティショナ／タイルコンフィギュアラが、フレーム内の他のチャネルと独立に各チャネルを区分する。

【０１５０】

次に、エンコーダは、異なるチャネルからのウィンドウをフレームのタイルにグループ化する（１２２０）。たとえば、エンコーダは、ウィンドウが同一の開始位置および同一の終了位置を有する場合に、異なるチャネルからのウィンドウを単一のタイルに置く。代替案では、エンコーダは、異なるチャネルのどの部分を一緒にタイルにグループ化するかを判定するのに、開始位置／終了位置以外の判断基準を使用するか、開始位置／終了位置に加えて判断基準を使用することができる。

【０１５１】

一実施形態では、エンコーダが、フレームに関するウィンドウ構成のセット（１２１０）の後に（それと独立に）、タイルのグループ化（１２２０）を実行する。他の実施形態では、エンコーダが、ウィンドウ構成をセットする（１２１０）のと同時にウィンドウを

タイルにグループ化し（１２２０）て、たとえば、時間相関を優先する（より長いウィンドウを使用する）か、チャネル相関を優先する（より多くのチャネルを単一のタイルに置く）か、強制的にウィンドウを特定のタイルの組にあてはめることによってタイルの個数を制御する。

【０１５２】

次に、エンコーダは、エンコードされたオーディオデータと共に出力するために、フレームのタイル構成情報を送る（１２３０）。たとえば、エンコーダのパーティショナ／タイルコンフィギュアラが、タイルサイズおよびタイルのチャネルメンバ情報をＭＵＸに送る。代替案では、エンコーダが、タイル構成を指定する他の情報を送る。一実施形態では、エンコーダが、タイルグループ化（１２２０）の後にタイル構成情報を送る（１２３０）。他の実施形態では、エンコーダが、これらの処置を同時に実行する。

【０１５３】

図１３に、特定のビットストリーム構文による、マルチチャネルオーディオのフレームに関してタイルを構成し、タイル構成情報を送る手法（１３００）を示す流れ図である。図１３には、情報をビットストリームに入れるためにエンコーダによって実行される手法（１３００）が示され、デコーダは、対応する手法（フラグを読み取る、特定のタイルに関する構成情報を得る、など）を実行して、ビットストリーム構文に従ってフレームのタイル構成情報を検索する。代替案では、デコーダおよびエンコーダが、図１３に示されたオプションの１つまたは複数に関する別の構文、たとえば、異なるフラグまたは異なる順序付けを使用する構文を使用する。

【０１５４】

エンコーダは、当初は、フレームのチャネルのどれもがウィンドウに分割されないかどうかを検査する（１３１０）。そうである場合には、エンコーダは、フラグビット（どのチャネルも分割されないことを示す）を送り（１３１２）、終了する。したがって、単一のビットによって、所与のフレームが単一のタイルであるか複数のタイルを有するかが示される。

【０１５５】

その一方で、少なくとも１つのチャネルがウィンドウに分割される場合に、エンコーダは、フレームのすべてのチャネルが同一のウィンドウ構成を有するか否かを検査する（１３２０）。そうである場合には、エンコーダは、フラグビット（すべてのチャネルが同一のウィンドウ構成を有し、フレームの各タイルがすべてのチャネルを有することを示す）とタイルサイズのシーケンスとを送り（１３２２）、終了する。したがって、単一のビットによって、チャネルのすべてが同一の構成を有する（通常のエンコーダビットストリームと同様に）か、柔軟なタイル構成を有するかが示される。

【０１５６】

少なくともいくつかのチャネルが異なるウィンドウ構成を有する場合に、エンコーダは、フレームのサンプル位置をスキャンして、同一の開始位置および同一の終了位置の両方を有するウィンドウを識別する。しかし、まず、エンコーダは、フレームのすべてのサンプル位置をグループ化されないものとしてマークする（１３３０）。次に、エンコーダは、チャネル／時間スキャンパターンに従って、フレームの次のグループ化されていないサンプル位置をスキャンする（１３４０）。一実施形態では、エンコーダが、グループ化されていないサンプル位置を探して特定の時刻のすべてのチャネルをスキャンし、その後、時間的に次のサンプル位置について繰り返す。他の実施形態では、エンコーダが、別のスキャンパターンを使用する。

【０１５７】

検出されたグループ化されていないサンプル位置について、エンコーダは、類似するウィンドウを一緒にタイルにグループ化する（１３５０）。具体的に言うと、エンコーダは、検出されたグループ化されていないサンプル位置を含むウィンドウの開始位置で始まり、検出されたグループ化されていないサンプル位置を含むウィンドウと同一の位置で終わるウィンドウをグループ化する。たとえば、図１１ｂに示されたフレームでは、エンコー

ダは、まず、チャネル０の先頭でサンプル位置を検出する。エンコーダは、チャネル０、２、３、および４からの１／４フレーム長のウィンドウを、一緒にタイルにグループ化する。というのは、これらのウィンドウのそれぞれが、タイルの他のウィンドウと同一の開始位置および同一の終了位置を有するからである。

【０１５８】

次に、エンコーダは、エンコードされたオーディオデータと共に出力するために、タイルを指定するタイル構成情報を送る（１３６０）。タイル構成情報には、タイルサイズと、タイル内のその点でグループ化されていないサンプル位置を有するどのチャネルがタイルに含まれるかを示すマップが含まれる。チャネルマップには、タイルに可能なチャネルごとに１ビットを含めることができる。タイル情報のシーケンスに基づいて、デコーダは、タイルがフレーム内で始まり、終わるかどうかを判定する。エンコーダは、どのチャネルがタイルに存在することができるかを考慮に入れることによって、チャネルのビットレートを下げる。たとえば、図１１ｂのタイル０の情報には、タイルサイズと、チャネル０、２、３、および４がタイルの一部であることを示すバイナリパターン「１０１１１０」が含まれる。その点の後で、チャネル１および５のサンプル位置だけが、グループ化されていない。したがって、タイル１の情報には、タイルサイズと、チャネル１がタイルの一部であるが、チャネル５がそうでないことを示すバイナリパターン「１０」が含まれる。これによって、バイナリパターンの４ビットが節約される。次に、タイル２のタイル情報に、タイルサイズだけが含まれる（チャネルマップは含まれない）。というのは、チャネル５が、タイル２で始まるウィンドウを有することができる唯一のチャネルであるからである。タイル３のタイル情報には、タイルサイズと、バイナリパターン「１１１１」が含まれる。というのは、チャネル１および５が、タイル３の範囲内のグループ化された位置を有するからである。代替案では、エンコーダおよびデコーダが、別の手法を使用して、構文でチャネルパターンを知らせる。

【０１５９】

次に、エンコーダは、タイルに含まれるウィンドウのサンプル位置を、グループ化されたものとしてマークし（１３７０）、継続するか否かを判定する（１３８０）。グループ化されていないサンプル位置がフレームにない場合には、エンコーダは終了する。そうでない場合には、エンコーダは、チャネル／時間スキャンパターンに従って、フレームの次のグループ化されていないサンプル位置をスキャンする（１３４０）。

【０１６０】

Ｖ．柔軟なマルチチャネル変換

いくつかの実施形態で、図６のエンコーダ（６００）などのエンコーダが、相互チャネル相関性を効果的に活用する柔軟なマルチチャネル変換を実行する。図７のデコーダ（７００）などのデコーダが、対応する逆マルチチャネル変換を実行する。

【０１６１】

具体的に言うと、エンコーダおよびデコーダは、下記の１つまたは複数を行って、異なる情況でマルチチャネル変換を改善する。

【０１６２】

１．エンコーダは、知覚的重みづけの後にマルチチャネル変換を実行し、デコーダは、逆重みづけの前に、対応する逆マルチチャネル変換を実行する。これによって、逆マルチチャネル変換後のチャネルにまたがる量子化ノイズのアンマスキングが減る。

【０１６３】

２．エンコーダおよびデコーダは、マルチチャネル変換のためにチャネルをグループ化して、どのチャネルが一緒に変換されるかを制限する。

【０１６４】

３．エンコーダおよびデコーダは、どの帯域が一緒に変換されるかを制御するために、周波数帯域レベルでマルチチャネル変換を選択的にオン／オフにする。

【０１６５】

４．エンコーダおよびデコーダは、階層マルチチャネル変換を使用して、計算の複雑さ

を（特にデコーダで）制限する。

【０１６６】

５．エンコーダおよびデコーダは、事前定義のマルチチャネル変換行列を使用して、変換行列の指定に使用されるビットレートを減らす。

【０１６７】

６．エンコーダおよびデコーダは、ビット効率のために、量子化されたギブンス回転ベースの因数分解パラメータを使用して、マルチチャネル変換行列を指定する。

【０１６８】

Ａ．重みづけされたマルチチャネルオーディオに対するマルチチャネル変換

いくつかの実施形態で、エンコーダは、知覚的重みづけの後にマルチチャネル変換を位置付け（デコーダは、逆重みづけの前に逆マルチチャネル変換を位置付け）、チャネル間の漏れ信号が、制御され、測定可能であり、元の信号に類似するスペクトルを有するようにする。

【０１６９】

図１４に、エンコーダで知覚的重みづけの後に１つまたは複数のマルチチャネル変換を実行する手法（１４００）を示す。エンコーダは、マルチチャネルオーディオに知覚的に重みをつけ（１４１０）、たとえば、周波数領域のマルチチャネルオーディオに重みづけ係数を適用する。いくつかの実施形態で、エンコーダは、マルチチャネル変換の前に、重みづけ係数とチャネルごとの量子化ステップ変更子の両方をマルチチャネルオーディオデータに適用する。

【０１７０】

次に、エンコーダは、たとえば下で説明するように、重みをつけられたオーディオデータに対する１つまたは複数のマルチチャネル変換を実行する（１４２０）。最後に、エンコーダは、マルチチャネル変換されたオーディオデータを量子化する（１４３０）。

【０１７１】

図１５に、デコーダで逆重みづけの前に逆マルチチャネル変換を実行する手法（１５００）を示す。デコーダは、たとえば下で説明するように、量子化されたオーディオデータに対して１つまたは複数の逆マルチチャネル変換を実行する（１５１０）。具体的に言うと、デコーダは、特定の周波数インデックスの複数のチャネルからのサンプルをベクトルx_{mc}に集め、逆マルチチャネル変換A_{mc}を実行して、出力y_{mc}を生成する。

$$y_{mc}=A_{mc}\cdot x_{mc} \qquad (7)$$

【０１７２】

その後、デコーダは、マルチチャネルオーディオを逆量子化し、逆重みづけし（１５２０）、マスクによって逆マルチチャネル変換の出力をカラーリングする。したがって、チャネルにまたがって発生する（量子化に起因する）漏れが、スペクトルにおいて整形され、その結果、漏れた信号の可聴性が、測定可能かつ制御可能であり、所与の再構成されたチャネルでの他のチャネルの漏れが、所与のチャネルの元の壊されない信号と同様にスペクトルにおいて整形される（いくつかの実施形態で、チャネルごとの量子化ステップサイズ変更子によって、エンコーダが、再構成される信号の品質がすべての再構成されるチャネルにまたがってほぼ同一になるようにすることを可能にすることもできる）。

【０１７３】

Ｂ．チャネルグループ

いくつかの実施形態で、エンコーダおよびデコーダが、マルチチャネル変換のためにチャネルをグループ化して、一緒に変換されるチャネルを制限する。たとえば、タイル構成を使用する実施形態では、エンコーダが、タイルのどのチャネルが相関するかを判定し、相関するチャネルをグループ化する。代替案では、エンコーダおよびデコーダが、タイル構成を使用しないが、フレームまたは他のレベルでチャネルをグループ化する。

【０１７４】

図１６に、一実施形態でマルチチャネル変換についてタイルのチャネルをグループ化する手法（１６００）を示す。この手法（１６００）では、エンコーダが、チャネルの信号

の間の対単位の相関ならびにいくつかの場合に帯域の間の相関を考慮する。代替案では、エンコーダが、マルチチャネル変換についてチャネルをグループ化する時に、他のおよび／または追加の要因を考慮する。

【０１７５】

まず、エンコーダは、タイルのチャネルを得る（１６１０）。たとえば、図１１ｂに示されたタイル構成では、タイル３が、その中に４つのチャネルすなわち０、２、３、および４を有する。

【０１７６】

エンコーダは、チャネルの信号の間の対単位の相関を計算し（１６２０）、それ相応にチャネルをグループ化する（１６３０）。図１１ｂのタイル３について、チャネル０および２が、対単位で相関するが、この両方のチャネルが、チャネル３またはチャネル４と対単位で相関せず、チャネル３が、対単位でチャネル４と相関しないと仮定する。エンコーダは、チャネル０および２を一緒にグループ化し（１６３０）、チャネル３を別のグループに入れ、チャネル４をさらに別のグループに入れる。

【０１７７】

グループのどのチャネルとも対単位で相関しないチャネルが、それでもそのグループとの互換性を有する場合がある。したがって、グループとの互換性がないチャネルについて、エンコーダは、任意選択として、帯域レベルでの互換性を検査し（１６４０）、それ相応にチャネルの１つまたは複数のグループを調整する（１６５０）。具体的に言うと、これによって、ある帯域でグループとの互換性があるが、他の帯域で非互換であるチャネルが識別される。たとえば、図１１ｂのタイル３のチャネル４が、実際にはほとんどの帯域でチャネル０および２と互換であるが、少数の帯域での非互換性のゆえに、対単位の相関結果が歪曲されると仮定する。エンコーダは、グループを調整して（１６５０）、チャネル０、２、および４を一緒にし、チャネル３をそれ自体のグループに残す。エンコーダは、いくつかのチャネルが「全体的に」相関するが、非互換帯域を有する時に、このようなテストを実行することもできる。これらの非互換帯域で変換をオフにすることによって、実際にマルチチャネル変換コーディングされる帯域の間の相関が改善され、したがって、コーディング効率が改善される。

【０１７８】

所与のタイルのチャネルは、１つのチャネルグループに属する。チャネルグループのチャネルが、連続的である必要はない。単一のタイルに、複数のチャネルグループを含めることができ、各チャネルグループが、異なる関連するマルチチャネル変換を有することができる。どのチャネルが互換性を有するかを判断した後に、エンコーダは、チャネルグループ情報をビットストリームに入れる。

【０１７９】

図１７に、エンコーダがチャネルグループを計算する方法に関係のない、特定のビットストリーム構文によるビットストリームからのタイルのチャネルグループ情報およびマルチチャネル変換情報の検索の手法（１７００）を示す。図１７には、ビットストリームから情報を検索するためにデコーダによって実行される手法（１７００）が示され、エンコーダは、対応する手法を実行して、ビットストリーム構文に従って、タイルのチャネルグループ情報およびマルチチャネル変換情報をフォーマットする。代替案では、デコーダおよびエンコーダが、図１７に示されたオプションの１つまたは複数について別の構文を使用する。

【０１８０】

まず、デコーダは、手法（１７００）で使用される複数の変数を初期化する。デコーダは、タイル＃ＣｈａｎｎｅｌｓＩｎＴｉｌｅのチャネル数と等しくなるように＃ＣｈａｎｎｅｌｓＴｏＶｉｓｉｔをセットし（１７１０）、チャネルグループ数＃ＣｈａｎｎｅｌＧｒｏｕｐｓに０をセットする（１７１２）。

【０１８１】

デコーダは、＃ＣｈａｎｎｅｌｓＴｏＶｉｓｉｔが２を超えるかどうかを検査する（１

７２０）。そうでない場合には、デコーダは、＃ＣｈａｎｎｅｌｓＴｏＶｉｓｉｔが２と等しいかどうかを検査する（１７３０）。そうである場合には、デコーダは、たとえば下で説明する手法を使用して、２チャネルのグループのマルチチャネル変換をデコードする（１７４０）。構文では、各チャネルグループが、異なるマルチチャネル変換を有することができる。その一方で、＃ＣｈａｎｎｅｌｓＴｏＶｉｓｉｔが１または０と等しい場合には、デコーダは、マルチチャネル変換をデコードせずに終了する。

【０１８２】

＃ＣｈａｎｎｅｌｓＴｏＶｉｓｉｔが２を超える場合には、デコーダは、タイルのグループのチャネルマスクをデコードする（１７５０）。具体的には、デコーダは、チャネルマスクのビットストリームから＃ＣｈａｎｎｅｌｓＴｏＶｉｓｉｔビットを読み取る。チャネルマスクの各ビットによって、特定のチャネルがチャネルグループに含まれるか否かが示される。たとえば、チャネルマスクが「１０１１０」である場合に、タイルに５つのチャネルが含まれ、チャネル０、２、および３がチャネルグループに含まれる。

【０１８３】

デコーダは、グループのチャネル数をカウントし（１７６０）、たとえば下で説明する手法を使用して、グループのマルチチャネル変換をデコードする（１７７０）。デコーダは、現在のチャネルグループのカウントされたチャネル数を引くことによって＃ＣｈａｎｎｅｌｓＴｏＶｉｓｉｔを更新し（１７８０）、＃ＣｈａｎｎｅｌＧｒｏｕｐｓを増分し（１７９０）、視察すべき残されたチャネル数＃ＣｈａｎｎｅｌｓＴｏＶｉｓｉｔが２を超えるかどうかを検査する（１７２０）。

【０１８４】

代替案では、タイル構成を使用しない実施形態で、デコーダが、フレームまたは他のレベルに関するチャネルグループ情報およびマルチチャネル変換情報を検索する。

【０１８５】

Ｃ．マルチチャネル変換の帯域オン／オフ制御

いくつかの実施形態で、エンコーダおよびデコーダが、周波数帯域レベルでマルチチャネル変換を選択的にオン／オフにして、どの帯域が一緒に変換されるかを制御する。この形で、エンコーダおよびデコーダが、マルチチャネル変換で互換性がない帯域を選択的に除外する。マルチチャネル変換が、特定の帯域についてオフにされる時に、エンコーダおよびデコーダは、その帯域に恒等変換を使用し、データを変更せずにその帯域のデータを通過させる。

【０１８６】

周波数帯域は、臨界帯域または量子化帯域である。周波数帯域の数は、オーディオデータのサンプリング周波数およびタイルサイズに関係する。一般に、サンプリング周波数が高くなるかタイルサイズが大きくなると、周波数帯域の数が増える。

【０１８７】

いくつかの実施形態で、エンコーダが、タイルのチャネルグループのチャネルについて、周波数帯域レベルでマルチチャネル変換を選択的にオン／オフにする。エンコーダは、タイルのチャネルをグループ化する時またはタイルに関するチャネルグループ化の後に、帯域をオン／オフにすることができる。代替案では、エンコーダおよびデコーダが、タイル構成を使用するのではなく、フレームまたは他のレベルについて周波数帯域でマルチチャネル変換をオン／オフにする。

【０１８８】

図１８に、一実施形態でマルチチャネル変換にチャネルグループのチャネルの周波数帯域を選択的に含める手法（１８００）を示す。手法（１８００）では、エンコーダが、帯域のチャネルの信号の間の対単位の相関を検討して、その帯域のマルチチャネル変換を使用可能にするか使用不能にするかを判定する。代替案では、エンコーダが、マルチチャネル変換について周波数帯域を選択的にオンまたはオフにする時に、他のおよび／または追加の要因を検討する。

【０１８９】

まず、エンコーダは、たとえば図１６に関して説明したように、チャネルグループのチャネルを入手する（１８１０）。次に、エンコーダは、異なる周波数帯域のチャネルの信号の間の対単位の相関を計算する（１８２０）。たとえば、チャネルグループに２つのチャネルが含まれる場合に、エンコーダは、各周波数帯域での対単位の相関を計算する。あるいは、チャネルグループに２つを超えるチャネルが含まれる場合に、エンコーダは、各周波数帯域でのめいめいのチャネル対の一部またはすべての間の対単位の相関を計算する。

【０１９０】

次に、エンコーダは、チャネルグループのマルチチャネル変換について、帯域をオンまたはオフにする（１８３０）。たとえば、チャネルグループに２つのチャネルが含まれる場合に、エンコーダは、帯域での対単位の相関が特定の閾値を満足する場合に、その帯域のマルチチャネル変換を使用可能にする。あるいは、チャネルグループに２つを超えるチャネルが含まれる場合に、エンコーダは、帯域の対単位の相関のそれぞれまたは大多数が特定の閾値を満足する場合に、その帯域のマルチチャネル変換を使用可能にする。代替実施形態では、すべてのチャネルについて特定の周波数帯域をオンまたはオフにするのではなく、エンコーダが、帯域を、あるチャネルについてオン、他のチャネルについてオフにする。

【０１９１】

どの帯域がマルチチャネル変換に含まれるかを判断した後に、エンコーダは、帯域オン／オフ情報をビットストリームに入れる。

【０１９２】

図１９に、エンコーダがどのように帯域をオンまたはオフにすると判断するかに無関係に、特定のビットストリーム構文によるビットストリームからのタイルのチャネルグループに関するマルチチャネル変換の帯域オン／オフ情報を検索する手法（１９００）を示す。図１９には、ビットストリームから情報を検索するためにデコーダによって実行される手法（１９００）が示され、エンコーダは、対応する手法を実行して、ビットストリーム構文に従ってチャネルグループの帯域オン／オフ情報をフォーマットする。代替案では、デコーダおよびエンコーダが、図１９に示されたオプションの１つまたは複数について別の構文を使用する。

【０１９３】

いくつかの実施形態で、デコーダは、手法（１７００）のマルチチャネル変換のデコード（１７４０または１７７０）の一部として手法（１９００）を実行する。代替案では、デコーダが、手法（１９００）を別々に実行する。

【０１９４】

デコーダは、ビットを入手し（１９１０）、ビットを検査して（１９２０）、チャネルグループについて、すべての帯域が使用可能にされているかどうかを判定する。そうである場合には、デコーダは、チャネルグループのすべての帯域についてマルチチャネル変換を使用可能にする（１９３０）。

【０１９５】

その一方で、ビットによって、チャネルグループのすべての帯域が使用可能にされているのでないことが示される場合に、デコーダは、チャネルグループの帯域マスクをデコードする（１９４０）。具体的に言うと、デコーダは、ビットストリームからビット数を読み取るが、この数は、チャネルグループの帯域の数である。帯域マスクの各ビットが、特定の帯域がチャネルグループについてオンまたはオフのどちらであるかを示す。たとえば、帯域マスクが「１１１１１１１１０１１００００」である場合には、チャネルグループに１５個の帯域が含まれ、帯域０、１、２、３、４、５、６、７、９、および１０が、マルチチャネル変換についてオンにされている。デコーダは、示された帯域についてマルチチャネル変換を使用可能にする（１９５０）。

【０１９６】

その代わりに、タイル構成を使用しない実施形態では、デコーダが、フレームまたは他

のレベルでの帯域オン／オフ情報を検索する。

【０１９７】

Ｄ．階層マルチチャネル変換

いくつかの実施形態で、エンコーダおよびデコーダは、階層マルチチャネル変換を使用して、特にデコーダでの、計算の複雑さを制限する。階層変換を用いるときに、エンコーダが、全体的な変換を複数のステージに分割し、個々のステージの計算の複雑さを減らし、いくつかの場合にマルチチャネル変換を指定するのに必要な情報の量を減らす。このカスケード構造を使用して、エンコーダは、より大きい全体的な変換を、ある精度まで、より小さい変換を用いてエミュレートする。デコーダは、対応する階層逆変換を実行する。

【０１９８】

いくつかの実施形態で、階層変換の各ステージが、構造において同一であり、ビットストリーム内で、各ステージが、１つまたは複数の他のステージと独立に記述される。具体的に言うと、各ステージが、それ自体のチャネルグループと、チャネルグループごとに１つのマルチチャネル変換行列を有する。代替実施形態では、異なるステージが、異なる構造を有し、エンコーダおよびデコーダで、異なるビットストリーム構文が使用され、かつ／またはステージで、チャネルおよび変換に関する別の構成が使用される。

【０１９９】

図２０に、より単純なマルチチャネル変換の階層を使用してマルチチャネル変換をエミュレートする一般化された手法（２０００）を示す。図２０には、ｎステージの階層が示され、ｎは、マルチチャネル変換ステージの数である。たとえば、一実施形態で、ｎは２である。代替案では、ｎが２より大きい。

【０２００】

エンコーダは、全体的な変換のマルチチャネル変換の階層を判定する（２０１０）。エンコーダは、逆変換を実行するデコーダの複雑さに基づいて、変換サイズ（すなわち、チャネルグループサイズ）を判断する。あるいは、エンコーダは、ターゲットデコーダプロファイル／デコーダレベルまたは他の判断基準を検討する。

【０２０１】

図２１は、マルチチャネル変換の例の階層（２１００）を示す図である。この階層（２１００）には、２つのステージが含まれる。第１ステージには、０からＮまでの番号をつけられた、Ｎ＋１個のチャネルグループおよび変換が含まれ、第２ステージには、０からＭまでの番号をつけられた、Ｍ＋１個のチャネルグループおよび変換が含まれる。各チャネルグループに、１つまたは複数のチャネルが含まれる。第１ステージのＮ＋１個の変換のそれぞれについて、入力チャネルは、マルチチャネルトランスフォーマに入力されるチャネルのある組合せである。すべての入力チャネルを第１ステージで変換しなければならないわけではない。１つまたは複数の入力チャネルを、無変更で第１ステージを通過させることができる（たとえば、エンコーダによって、チャネルグループに含まれる、単位行列を使用するチャネルを含めることができる）。第２ステージのＭ＋１個の変換のそれぞれについて、入力チャネルは、第１ステージからの出力チャネルのある組合せであり、この出力チャネルには、無変更で第１ステージを通過した可能性があるチャネルが含まれる。

【０２０２】

図２０に戻って、エンコーダは、マルチチャネル変換の第１ステージを実行し（２０２０）、マルチステージ変換の次のステージを実行し、最後に、マルチチャネル変換の第ｎステージを実行する（２０３０）。デコーダは、デコード中に、対応する逆マルチチャネル変換を実行する。

【０２０３】

いくつかの実施形態で、チャネルグループが、階層の複数のステージで同一であるが、マルチチャネル変換は異なる。そのような場合、およびいくつかの他の場合に、エンコーダは、複数のマルチチャネル変換について周波数帯域オン／オフ情報を組み合わせることができる。たとえば、２つのマルチチャネル変換があり、それぞれのチャネルグループに

同一の３つのチャネルがあると仮定する。エンコーダは、帯域０の両方のステージで変換なし／恒等変換、帯域１のマルチチャネル変換ステージ１のみ（ステージ２変換なし）、帯域２のマルチチャネル変換ステージ２のみ（ステージ１変換なし）、帯域３の両方のステージのマルチチャネル変換、帯域４の両方のステージでの変換なしなどを指定することができる。

【０２０４】

図２２に、特定のビットストリーム構文によるビットストリームからのチャネルグループに関するマルチチャネル変換の階層の情報を検索する手法（２２００）を示す。図２２には、ビットストリームを解析するためにデコーダによって実行される手法（２２００）が示され、エンコーダは、対応する手法を実行して、ビットストリーム構文に従ってマルチチャネル変換の階層をフォーマットする。代替案では、デコーダおよびエンコーダが、別の構文、たとえば、２つを超えるステージのための追加フラグおよびシグナリングビットを含む構文を使用する。

【０２０５】

デコーダは、まず、ビットストリームの次のビットと等しくなるように一時値ｉＴｍｐをセットする（２２１０）。次に、デコーダは、一時値の値を検査し（２２２０）、この値によって、デコーダが、ステージ１グループのチャネルグループおよびマルチチャネル変換情報をデコード（２２３０）しなければならないか否かが知らされる。

【０２０６】

デコーダは、ステージ１グループのチャネルグループおよびマルチチャネル変換情報をデコード（２２３０）した後に、ビットストリームの次のビットと等しくなるようにｉＴｍｐをセットする（２２４０）。デコーダは、ｉＴｍｐの値を検査する（２２２０）が、この値によって、さらなるステージ１グループに関するチャネルグループおよびマルチチャネル変換情報がビットストリームに含まれるか否かが知らされる。恒等変換を有しないチャネルグループだけが、ビットストリームのステージ１部分で指定され、ビットストリームのステージ１部分に記載されていないチャネルは、恒等変換を使用するチャネルグループの一部と仮定される。

【０２０７】

ビットストリームに、ステージ１グループのチャネルグループおよびマルチチャネル変換情報がこれ以上含まれない場合には、デコーダは、すべてのステージ２グループのチャネルグループおよびマルチチャネル変換情報をデコードする（２２５０）。

【０２０８】

Ｅ．事前定義またはカスタムのマルチチャネル変換

いくつかの実施形態で、エンコーダおよびデコーダが、事前定義のマルチチャネル変換行列を使用して、変換行列の指定に使用されるビットレートを減らす。エンコーダは、複数の使用可能な事前定義の行列タイプの中から選択し、選択された行列を、ビットストリーム内で少数（たとえば、１、２）のビットを使用して知らせる。行列のタイプの中には、ビットストリーム内の追加シグナリングを必要としないものと、追加の指定を必要とするものがある。デコーダは、行列のタイプを示す情報と（必要な場合に）行列を指定する追加情報を検索する。

【０２０９】

いくつかの実施形態で、エンコーダおよびデコーダが、下記の事前定義行列タイプを使用する：恒等、アダマール、ＤＣＴタイプＩＩ、または任意のユニタリ。代替案では、エンコーダおよびデコーダが、異なるおよび／または追加の事前定義行列タイプを使用する。

【０２１０】

図９ａに、別のコンテキストでの６つのチャネルの単位行列の例が示されている。単位行列の次元の数が、他の情報（たとえば、グループのチャネル数）からエンコーダおよびデコーダに既知になると仮定して、エンコーダは、フラグビットを使用して、ビットストリームで単位行列を効率的に指定する。

【０２１１】
アダマール行列は、下記の形を有する。
【０２１２】
【数６】

$$A_{Hadamard} = \rho \begin{bmatrix} 0.5 & -0.5 \\ 0.5 & 0.5 \end{bmatrix} \quad (8)$$

【０２１３】
ここで、ρは、正規化スケーラ
【０２１４】
【数７】

$$\left(\sqrt{2}\right)$$

【０２１５】
である。エンコーダは、ステレオデータのアダマール行列を、ビットストリーム内でフラグビットを使用して効率的に指定する。
【０２１６】
ＤＣＴタイプＩＩ行列は、下記の形を有する。
【０２１７】
【数８】

$$A_{DCT\text{-}II} = \begin{bmatrix} a_{0,0} & a_{0,1} & \cdots & a_{0,N-1} \\ a_{1,0} & a_{1,1} & \cdots & a_{1,N-1} \\ \cdots & \cdots & \cdots & \cdots \\ a_{N-1,0} & a_{N-1,1} & \cdots & a_{N-1,N-1} \end{bmatrix} \quad (9)$$

【０２１８】
ここで
【０２１９】
【数９】

$$a_{n,m} = k_m \cdot \cos\left(\frac{m \cdot (n+0.5)\pi}{N}\right) \quad (10)$$

【０２２０】
また、
【０２２１】
【数１０】

$$k_m = \begin{cases} \sqrt{\frac{1}{N}} & m = 0 \\ \sqrt{\frac{2}{N}} & m > 0 \end{cases} \quad (11)$$

【０２２２】
である。
【０２２３】
ＤＣＴタイプＩＩ行列に関する追加情報については、文献を参照されたい（たとえば、非特許文献４参照）。ＤＣＴタイプＩＩ行列は、任意のサイズを有することができる（すなわち、すべてのサイズのチャネルグループについて働く）。ＤＣＴタイプＩＩ行列の次元の数が、他の情報（たとえば、グループのチャネル数）からエンコーダおよびデコーダに既知になると仮定して、エンコーダは、フラグビットを使用して、ビットストリームで

単位行列を効率的に指定する。

【０２２４】

正方行列A_{square}は、その転置行列が逆行列である場合に、ユニタリである。

$$A_{square} \cdot A_{square}^{T} = A_{square}^{T} \cdot A_{square} = I \quad (12)$$

ここで、Ｉは、単位行列である。エンコーダは、任意のユニタリ行列を使用して、効果的な冗長性除去のためのＫＬＴ変換を指定する。エンコーダは、ビットストリーム内で、フラグビットおよび行列のパラメータ化を使用して、任意のユニタリ行列を効率的に指定する。いくつかの実施形態で、エンコーダは、下で説明するように、量子化されたギブンス因数分解回転を使用して行列をパラメータ化する。代替案では、エンコーダが、別のパラメータ化を使用する。

【０２２５】

図２３に、複数の使用可能なタイプの中からマルチチャネル変換タイプを選択する手法（２３００）を示す。エンコーダは、チャネルグループごとにまたはある他のレベルで、変換タイプを選択する。

【０２２６】

エンコーダは、複数の使用可能なタイプの中からマルチチャネル変換タイプを選択する（２３１０）。たとえば、使用可能なタイプに、恒等、アダマール、ＤＣＴタイプＩＩ、および任意のユニタリが含まれる。代替案では、タイプに、異なるおよび／または追加の行列タイプが含まれる。エンコーダは、可能な場合、または変換行列を指定するのに必要なビット数を減らすのに必要な場合に、恒等行列、アダマール行列、またはＤＣＴタイプＩＩ行列（任意のユニタリ行列ではなく）を使用する。たとえば、エンコーダは、冗長性除去が、任意のユニタリ行列による冗長性除去に匹敵するか十分に近い（ある判断基準によって）場合に、恒等行列、アダマール行列、またはＤＣＴタイプＩＩ行列を使用する。あるいは、エンコーダは、ビットレートを削減しなければならない場合に、恒等行列、アダマール行列、またはＤＣＴタイプＩＩ行列を使用する。しかし、一般的な情況で、エンコーダは、最良の圧縮効率のために任意のユニタリ行列を使用する。

【０２２７】

エンコーダは、選択されたタイプのマルチチャネル変換を、マルチチャネルオーディオデータに適用する（２３２０）。

【０２２８】

図２４に、複数の使用可能なタイプの中からマルチチャネル変換タイプを検索し、逆マルチチャネル変換を実行する手法（２４００）を示す。デコーダは、チャネルグループごとまたは他のレベルで変換タイプ情報を検索する。

【０２２９】

デコーダは、複数の使用可能なタイプの間からマルチチャネル変換タイプを検索する（２４１０）。たとえば、使用可能なタイプに、恒等、アダマール、ＤＣＴタイプＩＩ、および任意のユニタリが含まれる。代替案では、タイプに、異なるおよび／または追加の行列タイプが含まれる。必要な場合には、デコーダは、行列を指定する追加情報を検索する。

【０２３０】

行列を再構成した後に、デコーダは、選択されたタイプの逆マルチチャネル変換をマルチチャネルオーディオデータに適用する（２４２０）。

【０２３１】

図２５に、特定のビットストリーム構文によるビットストリームからチャネルグループに関するマルチチャネル変換情報を検索する手法（２５００）を示す。図２５には、ビットストリームを解析するためにデコーダによって実行される手法（２５００）が示されているが、エンコーダは、対応する手法を使用して、ビットストリーム構文に従ってマルチチャネル変換情報をフォーマットする。代替案では、デコーダおよびエンコーダが、別の構文、たとえば、異なるフラグビット、異なる順序付け、または異なる変換タイプを使用する構文を使用する。

【０２３２】
当初、デコーダは、グループのチャネル数＃ＣｈａｎｎｅｌｓＩｎＧｒｏｕｐが１より大きいかどうかを検査する（２５１０）。そうでない場合には、チャネルグループがモノラルオーディオであり、デコーダは、グループに恒等変換を使用する（２５１２）。
【０２３３】
＃ＣｈａｎｎｅｌｓＩｎＧｒｏｕｐが１より大きい場合には、デコーダは、＃ＣｈａｎｎｅｌｓＩｎＧｒｏｕｐが２より大きいかどうかを検査する（２５２０）。そうでない場合には、チャネルグループはステレオオーディオであり、デコーダは、ビットストリームの次のビットと等しくなるように一時値ｉＴｍｐをセットする（２５２２）。次に、デコーダは、一時値の値を検査する（２５２４）が、この値によって、デコーダが、そのチャネルグループにアダマール変換を使用（２５３０）しなければならないかどうかが示される。そうでない場合には、デコーダは、ビットストリームの次のビットと等しくなるように一時値ｉＴｍｐをセットし（２５２６）、ｉＴｍｐの値を検査し（２５２８）、この値によって、デコーダが、チャネルグループに恒等変換を使用（２５５０）しなければならないかどうかが示される。そうでない場合には、デコーダは、チャネルグループに汎用ユニタリ変換をデコードする（２５７０）。
【０２３４】
＃ＣｈａｎｎｅｌｓＩｎＧｒｏｕｐが２より大きい場合には、チャネルグループは、サラウンドサウンドオーディオであり、デコーダは、ビットストリームの次のビットと等しくなるように一時値ｉＴｍｐをセットする（２５４０）。デコーダは、一時値の値を検査し（２５４２）、この値によって、デコーダが、チャネルグループのサイズ＃ＣｈａｎｎｅｌｓＩｎＧｒｏｕｐの恒等変換を使用（２５５０）しなければならないかどうかが示される。そうでない場合には、デコーダは、ビットストリームの次のビットと等しくなるように一時値ｉＴｍｐをセットし（２５６０）、ｉＴｍｐの値を検査する（２５６２）。このビットによって、デコーダが、チャネルグループの汎用ユニタリ変換をデコード（２５７０）しなければならないか、チャネルグループのサイズ＃ＣｈａｎｎｅｌｓＩｎＧｒｏｕｐのＤＣＴタイプＩＩ変換を使用（２５８０）しなければならないかが示される。
【０２３５】
デコーダは、チャネルグループに関してアダマール変換行列、ＤＣＴタイプＩＩ変換行列、または汎用ユニタリ変換行列を使用する時に、行列のマルチチャネル変換帯域オン／オフ情報をデコードし（２５９０）、終了する。
【０２３６】
Ｆ．変換行列のギブンス回転表現
いくつかの実施形態で、エンコーダおよびデコーダが、ビット効率のために、量子化されたギブンス回転ベースの因数分解パラメータを使用して、任意のユニタリ変換行列を指定する。
【０２３７】
一般に、ユニタリ変換行列は、ギブンス因数分解回転を使用して表すことができる。この因数分解を使用すると、ユニタリ変換行列を、次のように表すことができる。
【０２３８】
【数１１】

$$A_{\text{unitary}} = \Theta_{0,N-2} \cdots \Theta_{0,1} \Theta_{0,0} \Theta_{1,N-3} \cdots \Theta_{1,1} \Theta_{1,0} \cdots \Theta_{N-2,0} \begin{bmatrix} \alpha_0 & 0 & \cdots & 0 \\ 0 & \alpha_1 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & \alpha_{N-1} \end{bmatrix} \qquad (13)$$

【０２３９】
ここで、α_iは、＋１または－１（回転の符号）であり、各Θは、図２６に示された回転行列（２６００）の形である。回転行列（２６００）は、単位行列にほとんど似ているが

、変化する位置に４つのサイン／コサイン項を有する。図２７ａから２７ｃに、マルチチャネル変換行列を表すギブンス回転の例の回転行列を示す。２つのコサイン項が、必ず対角線上にあり、２つのサイン項が、コサイン項と同一の行／列にある。各Θは、１つの回転角度を有し、その値は、範囲

【０２４０】

【数１２】

$$-\frac{\pi}{2} \le \omega_k < \frac{\pi}{2}$$

【０２４１】

を有することができる。Ｎ×Ｎユニタリ行列$A_{unitary}$を完全に記述するのに必要なそのような回転行列Θの数は、次の通りである。

【０２４２】

【数１３】

$$\frac{N(N-1)}{2} \qquad (14)$$

【０２４３】

ギブンス因数分解回転に関する追加情報については、参照によって本明細書に組み込まれる文献を参照されたい（たとえば、非特許文献５参照）。

【０２４４】

いくつかの実施形態で、エンコーダは、ギブンス因数分解の回転角度を量子化して、ビットレートを減らす。図２８に、量子化されたギブンス因数分解回転を使用してマルチチャネル変換行列を表す手法（２８００）を示す。代替案では、エンコーダまたは処理ツールが、量子化されたギブンス因数分解回転を使用して、オーディオチャネルのマルチチャネル変換以外の目的のユニタリ行列を表す。

【０２４５】

エンコーダは、まず、マルチチャネル変換の任意のユニタリ行列を計算する（２８１０）。次に、エンコーダは、ユニタリ行列のギブンス因数分解回転を計算する（２８２０）。

【０２４６】

ビットレートを減らすために、エンコーダは、回転角度を量子化する（２８３０）。一実施形態では、エンコーダが、各回転角度を６４個（$2^6 = 64$）の可能な値の１つに均等に量子化する。回転の符号は、それぞれ１ビットによって表され、したがって、エンコーダは、下記の数のビットを使用して、Ｎ×Ｎユニタリ行列を表す。

【０２４７】

【数１４】

$$6 \cdot \frac{N(N-1)}{2} + N = 3N^2 - 2N \qquad (15)$$

【０２４８】

このレベルの量子化を用いると、エンコーダが、非常によい度合の精度で、マルチチャネル変換のＮ×Ｎユニタリ行列を表せるようになる。代替案では、エンコーダが、ある他のレベルおよび／またはタイプの量子化を使用する。

【０２４９】

図２９に、特定のビットストリーム構文によるビットストリームからチャネルグループの汎用ユニタリ変換の情報を検索する手法（２９００）を示す。図２９には、ビットストリームを解析するためにデコーダによって実行される手法（２９００）が示され、エンコーダは、対応する手法を実行して、ビットストリーム構文に従って汎用ユニタリ変換の情報をフォーマットする。代替案では、デコーダおよびエンコーダが、別の構文、たとえば

、異なる順序付けまたは回転角度の分解能を使用する構文を使用する。

【０２５０】

まず、デコーダは、デコードの残りで使用される複数の変数を初期化する。具体的に言うと、デコーダは、デコードする角度の数＃ＡｎｇｌｅｓＴｏＤｅｃｏｄｅを、式１４に示されたチャネルグループのチャネルの数＃ＣｈａｎｎｅｌｓＩｎＧｒｏｕｐに基づいてセットする（２９１０）。デコーダは、＃ＣｈａｎｎｅｌｓＩｎＧｒｏｕｐに基づいて、デコードする符号の数＃ＳｉｇｎｓＴｏＤｅｃｏｄｅもセットする（２９１２）。デコーダは、デコードされた角度のカウンタｉＡｎｇｌｅｓＤｅｃｏｄｅｄおよびデコードされた符号のカウンタｉＳｉｇｎｓＤｅｃｏｄｅｄもリセットする（２９１４、２９１６）。

【０２５１】

デコーダは、デコードする角度があるかどうかを検査し（２９２０）、そうである場合には、次の回転角度の値をセットし（２９２２）、６ビットの量子化された値から回転角度を再構成する。

$$\mathrm{RotationAngle}[\mathrm{iAnglesDecoded}] = \pi \times (\mathrm{getBits}(6) - 32) / 64 \qquad (16)$$

【０２５２】

次に、デコーダは、デコードされた角度のカウンタを増分し（２９２４）、さらにデコードする追加の角度があるかどうかを検査する（２９２０）。

【０２５３】

デコードする角度がもうない時に、デコーダは、デコードする追加の符号があるかどうかを検査し（２９４０）、そうである場合には、次の符号の値をセットし（２９４２）、１ビットの値から符号を再構成する。

$$\mathrm{RotationSign}[\mathrm{iSignsDecoded}] = (2 \times \mathrm{getBits}(1)) - 1 \qquad (17)$$

【０２５４】

次に、デコーダは、デコードされた符号のカウンタを増分し（２９４４）、デコードする追加の符号があるかどうかを検査する（２９４０）。デコードする符号がもうない時に、デコーダは終了する。

【０２５５】

ＶＩ．量子化および重みづけ

いくつかの実施形態で、図６のエンコーダ（６００）などのエンコーダが、下で説明するさまざまな手法を使用して、オーディオデータに対する量子化および重みづけを実行する。タイルに構成されたマルチチャネルオーディオに関して、エンコーダは、タイルのチャネルの量子化行列、チャネルごとの量子化ステップ変更子、および全体的な量子化タイル係数を計算し、適用する。これによって、エンコーダが、聴覚モデルに従って雑音を整形し、チャネルの間の雑音のバランスをとり、全体的なひずみを制御できるようになる。

【０２５６】

図７のデコーダ（７００）などの対応するデコーダは、逆量子化および逆重みづけを実行する。タイルに構成されたマルチチャネルオーディオについて、デコーダは、全体的な量子化タイル係数、チャネルごとの量子化ステップ変更子、およびタイルのチャネルの量子化行列をデコードし、適用する。逆量子化および逆重みづけが、単一のステップに融合される。

【０２５７】

Ａ．全体的なタイル量子化係数

いくつかの実施形態で、タイルのオーディオデータの品質および／またはビットレートを制御するために、エンコーダのクォンタイザが、タイルの量子化ステップサイズQ_tを計算する。クォンタイザは、レート／品質コントローラと共に働いて、ビットレートおよび／または品質制約を満足するタイル量子化ステップサイズを選択する前に、タイルの異なる量子化ステップサイズを評価することができる。たとえば、クォンタイザおよびコントローラは、参照によって本明細書に組み込まれる関連特許出願の発明の名称"Quality a

nd Rate Control Strategy for Digital Audio," の米国特許出願第１０／０１７，６９４号（２００１年１２月１４日出願）に記載されているように動作する。

【０２５８】

図３０に、特定のビットストリーム構文によるビットストリームから全体的なタイル量子化係数を検索する手法（３０００）を示す。図３０には、ビットストリームを解析するためにデコーダによって実行される手法（３００）が示され、エンコーダは、対応する手法を実行して、ビットストリーム構文に従ってタイル量子化係数をフォーマットする。代替案では、デコーダおよびエンコーダが、別の構文、たとえば、タイル量子化係数の異なる範囲を扱うもの、異なる論理を使用してタイル係数をエンコードするもの、またはタイル係数のグループをエンコードするものを使用する。

【０２５９】

まず、デコーダは、タイルの量子化ステップサイズQ_tを初期化する（３０１０）。一実施形態では、デコーダは、Q_tに下記をセットする。

$$Q_t = 90 \cdot ValidBitsPerSample / 16 \qquad (18)$$

ここで、ＶａｌｉｄＢｉｔｓＰｅｒＳａｍｐｌｅは、１６≦ＶａｌｉｄＢｉｔｓＰｅｒＳａｍｐｌｅ≦２４の数であり、デコーダまたはオーディオクリップについてセットされるか、他のレベルでセットされる。

【０２６０】

次に、デコーダは、Q_tの初期値に関するQ_tの最初の修正を示す６ビットを入手し（３０２０）、値－３２≦Ｔｍｐ≦３１を一時変数Ｔｍｐに保管する。関数ＳｉｇｎＥｘｔｅｎｄ（）は、符号なしの値から符号付きの値を判定する。デコーダは、Ｔｍｐの値をQ_tの初期値に加算し（３０３０）、その後、変数Ｔｍｐの符号を判定し（３０４０）、この符号は、変数ＳｉｇｎｏｆＤｅｌｔａに保管される。

【０２６１】

デコーダは、Ｔｍｐの値が－３２または３１と等しいかどうかを検査する（３０５０）。そうでない場合には、デコーダは終了する。Ｔｍｐの値が－３２または３１と等しい場合には、エンコーダは、Q_tをさらに修正しなければならないことを知らされている。さらなる修正の方向（正または負）は、ＳｉｇｎｏｆＤｅｌｔａによって示され、デコーダは、次の５ビットを得て（３０６０）、次の修正の大きさ０≦Ｔｍｐ≦３１を判定する。デコーダは、Q_tの現在の値を、ＳｉｇｎｏｆＤｅｌｔａの方向でＴｍｐの値だけ変更し（３０７０）、Ｔｍｐの値が３１であるかどうかを検査する（３０８０）。そうでない場合には、デコーダは終了する。Ｔｍｐの値が３１である場合には、デコーダは、次の５ビットを得て（３０６０）、その点から継続する。

【０２６２】

タイル構成を使用しない実施形態では、エンコーダが、フレームまたはオーディオデータの他の部分に関する全体的な量子化ステップサイズを計算する。

【０２６３】

Ｂ．チャネルごとの量子化ステップ変更子

いくつかの実施形態で、エンコーダは、タイルの各チャネルの量子化ステップ変更子：$Q_{c,0}$、$Q_{c,1}$、・・・、$Q_{c,\#ChannelsInTile-1}$を計算する。エンコーダは、通常は、これらのチャネル固有量子化係数を計算して、すべてのチャネルにまたがる再構成品質のバランスをとる。タイル構成を使用しない実施形態であっても、エンコーダは、フレームまたはオーディオデータの他の単位でチャネルのチャネルごとの量子化係数を計算することができる。対照的に、図１のエンコーダ（１００）で使用されるものなどの、以前の量子化手法は、チャネル内のウィンドウの帯域ごとに量子化行列要素を使用するが、チャネルに関する全体的な変更子を有しない。

【０２６４】

図３１に、マルチチャネルオーディオデータのチャネルごとの量子化ステップ変更子を計算する一般化された手法（３１００）を示す。エンコーダは、複数の判断基準を使用して、量子化ステップ変更子を計算する。第１に、エンコーダは、再構成されるオーディオ

データのすべてのチャネルにまたがってほぼ等しい品質を探す。第２に、スピーカ位置が既知である場合に、エンコーダは、スピーカ構成に関する通常の使用での知覚に最も重要なスピーカを優先する。第３に、スピーカタイプが既知である場合に、エンコーダは、スピーカ構成でのよりよいスピーカを優先する。代替案では、エンコーダが、これらの判断基準以外のまたはこれらの判断基準に加えて判断基準を考慮する。

【０２６５】

エンコーダは、チャネルの量子化ステップ変更子をセットする（３１１０）ことによって開始する。一実施形態では、エンコーダは、めいめいのチャネルのエネルギに基づいて変更子をセットする（３１１０）。たとえば、他のチャネルより相対的により多くのエネルギ（すなわち大音量）を有するチャネルについて、他のチャネルの量子化ステップ変更子が、比較的に大きくされる。代替案では、エンコーダが、「オープンループ」推定処理で、他のまたは追加の判断基準に基づいて変更子をセットする（３１１０）。あるいは、エンコーダは、変更子に当初は等しい値をセットする（３１１０）ことができる（変更子の最終的な値に集束するのに「クローズドループ」評価に頼ってセットする）。

【０２６６】

エンコーダは、量子化ステップ変更子ならびに、他の量子化（重みづけを含む）要因がまだ適用されていない場合にはそのような他の要因を使用して、マルチチャネルオーディオデータを量子化する（３１２０）。

【０２６７】

後続の再構成の後に、エンコーダは、ＮＥＲまたは他の品質測定値を使用して、再構成されたオーディオのチャネルの品質を評価する（３１３０）。エンコーダは、再構成されたオーディオが品質判断基準（および／または他の判断基準）を満足するか否かを検査し（３１４０）、そうである場合には終了する。そうでない場合には、エンコーダは、量子化ステップ変更子の新しい値をセットし（３１１０）、評価された結果に鑑みて変更子を調節する。代替案では、ステップ変更子の１パスのオープンループ設定について、エンコーダが、評価（３１３０）および検査（３１４０）をスキップする。

【０２６８】

チャネルごとの量子化ステップ変更子は、ウィンドウ／タイルからウィンドウ／タイルへと変化する傾向を有する。エンコーダは、リテラルまたは可変長コードとして量子化ステップ変更子をコーディングし、それをオーディオデータと共にビットストリームにパックする。あるいは、エンコーダは、他の手法を使用して、量子化ステップ変更子を処理する。

【０２６９】

図３２に、特定のビットストリーム構文によるビットストリームからチャネルごとの量子化ステップ変更子を検索する手法（３２００）を示す。図３２には、ビットストリームを解析するためにデコーダによって実行される手法（３２００）が示され、エンコーダは、対応する手法（フラグの設定、量子化ステップ変更子のデータのパックなど）を実行して、ビットストリーム構文に従って量子化ステップ変更子をフォーマットする。代替案では、デコーダおよびエンコーダが、別の構文、たとえば、異なるフラグまたは論理を処理して量子化ステップ変更子をエンコードする構文を使用する。

【０２７０】

図３２に、タイルのチャネルごとの量子化ステップ変更子の検索を示す。その代わりに、タイルを使用しない実施形態で、デコーダが、フレームまたはオーディオデータの他の単位に関してチャネルごとのステップ変更子を検索する。

【０２７１】

まず、デコーダは、タイルのチャネル数が１を超えるかどうかを検査する（３２１０）。そうでない場合には、オーディオデータがモノラルである。デコーダは、モノラルチャネルの量子化ステップ変更子に０をセットし（３２１２）、終了する。

【０２７２】

マルチチャネルオーディオについて、デコーダは、複数の変数を初期化する。デコーダ

は、タイルの量子化ステップ変更子ごとのビット数を示すビット（＃ＢｉｔｓＰｅｒＱ）を得る（３２２０）。一実施形態では、デコーダが、３ビットを得る。デコーダは、チャネルカウンタｉＣｈａｎｎｅｌｓＤｏｎｅに０をセットする（３２２２）。

【０２７３】

デコーダは、チャネルカウンタがタイルのチャネル数より少ないかどうかを検査する（３２３０）。そうでない場合には、タイルのすべてのチャネル量子化ステップ変更子が検索されており、デコーダは終了する

その一方で、チャネルカウンタが、タイルのチャネル数より少ない場合には、デコーダは、１ビットを入手し（３２３２）、そのビットを検査して（３２４０）、現在のチャネルの量子化ステップ変更子が０であるかどうかを判定する。そうである場合には、デコーダは、現在のチャネルの量子化ステップ変更子に０をセットする（３２４２）。

【０２７４】

現在のチャネルの量子化ステップ変更子が０でない場合には、デコーダは、＃ＢｉｔｓＰｅｒＱが０より大きいかどうかを検査して（３２５０）、現在のチャネルの量子化ステップ変更子が１であるかどうかを判定する。そうである場合には、デコーダは、現在のチャネルの量子化ステップ変更子に１をセットする（３２５２）。

【０２７５】

＃ＢｉｔｓＰｅｒＱが０より大きい場合には、デコーダは、ビットストリームの次の＃ＢｉｔｓＰｅｒＱビットを入手し、１を加算し（０の値がより以前の終了条件をトリガするので）、現在のチャネルの量子化ステップ変更子にその結果をセットする（３２６０）。

【０２７６】

デコーダは、現在のチャネルの量子化ステップ変更子をセットした後に、チャネルカウンタを増分し（３２７０）、チャネルカウンタがタイルのチャネル数より少ないかどうかを検査する（３２３０）。

【０２７７】

Ｃ．量子化行列のエンコーディングおよびデコーディング

いくつかの実施形態で、エンコーダは、タイルの各チャネルの量子化行列を計算する。エンコーダは、複数の形で、図１のエンコーダ（１００）で使用されるものなどの以前の量子化手法より改善される。量子化行列のロッシイ圧縮に関して、エンコーダは、量子化行列要素の柔軟なステップサイズを使用し、これによって、エンコーダが、量子化行列の要素の分解能を変更できるようになる。この特徴とは別に、エンコーダは、量子化行列の圧縮中に量子化行列値の時間的相関を活用する。

【０２７８】

前に述べたように、量子化行列は、タイルのチャネルごとに、バーク周波数帯域（または他の区分された量子化帯域）ごとに１ステップ値の、ステップサイズ配列として働く。エンコーダは、量子化行列を使用して、元の信号に匹敵するスペクトル形状を有するように、再構成されるオーディオ信号を「カラーリング」する。エンコーダは、通常は、音響心理学に基づいて量子化行列を判定し、量子化行列を圧縮して、ビットレートを下げる。量子化行列の圧縮は、ロッシイとすることができる。

【０２７９】

このセクションに記載の手法は、タイルのチャネルに関する量子化行列に関して説明される。表記について、$Q_{m,iChannel,iBand}$が、帯域ｉＢａｎｄのチャネルｉＣｈａｎｎｅｌの量子化行列要素を表すものとする。タイル構成を使用しない実施形態では、エンコーダが、量子化行列要素の柔軟なステップサイズを使用し、かつ／または、圧縮中の量子化行列値の時間的相関を活用することができる。

【０２８０】

１．マスク情報の柔軟な量子化ステップサイズ

図３３に、量子化行列要素の量子化ステップサイズを適応式にセットする一般化された手法（３３００）を示す。これによって、エンコーダが、マスク情報を粗くまたは微細に

量子化できるようになる。一実施形態では、エンコーダが、タイルのチャネルごとに（すなわち、タイルの各チャネルが行列を有する時には行列ごとに）量子化行列要素の量子化ステップサイズをセットする。代替案では、エンコーダが、オーディオシーケンス全体または他のレベルで、タイルごとにまたはフレームごとに、マスク要素の量子化ステップサイズをセットする。

【０２８１】

エンコーダは、１つまたは複数のマスクの量子化ステップサイズをセットする（３３１０）ことによって開始する（影響されるマスクの数は、エンコーダが柔軟な量子化ステップサイズを割り当てるレベルに依存する）。一実施形態では、エンコーダが、ある時間の期間にわたって再構成されるオーディオの品質を評価し、その結果に応じて、マスク情報の量子化ステップサイズを１ｄＢ、２ｄＢ、３ｄＢ、または４ｄＢになるように選択する。エンコーダによって評価される品質測定値は、１つまたは複数の前にエンコードされたフレームのＮＥＲである。たとえば、全体的な品質が低い場合に、エンコーダは、マスク情報の量子化ステップサイズにより高い値をセットする（３３１０）ことができる。というのは、量子化行列の分解能が、ビットレートの効率的な使用になっていないからである。その一方で、全体的な品質がよい場合に、エンコーダは、マスク情報の量子化ステップサイズにより低い値をセットする（３３１０）ことができる。というのは、量子化行列のよりよい分解能によって、知覚される品質が効率的に改善される可能性があるからである。代替案では、エンコーダが、量子化ステップサイズのオープンループ推定で、別の品質測定値、異なる期間にわたる評価、および／または他の判断基準を使用する。エンコーダは、マスク情報に異なるまたは追加の量子化ステップサイズを使用することもできる。あるいは、エンコーダが、オープンループ推定をスキップし、その代わりに、ステップサイズの最終的な値に集束するのに結果のクローズドループ評価に頼る。

【０２８２】

エンコーダは、マスク要素の量子化ステップサイズを使用して１つまたは複数の量子化行列を量子化し（３３２０）、マルチチャネルオーディオデータに重みをつけ、量子化する。

【０２８３】

後続の再構成の後に、エンコーダは、ＮＥＲまたは他の品質測定値を使用して、再構成されたオーディオの品質を評価する（３３３０）。エンコーダは、再構成されたオーディオの品質が、マスク情報に関する現在の量子化ステップサイズの設定を正当化するものであるかどうかを検査する（３３４０）。そうでない場合には、エンコーダは、マスク情報の量子化ステップサイズにより高いかより低い値をセットする（３３１０）ことができる。それ以外の場合には、エンコーダは終了する。代替案では、マスク情報の量子化ステップサイズの１パスオープンループ設定について、エンコーダが、評価（３３３０）および検査（３３４０）をスキップする。

【０２８４】

選択の後に、エンコーダは、ビットストリームの適当なレベルでマスク情報の量子化ステップサイズを示す。

【０２８５】

図３４に、量子化行列要素の適応量子化ステップサイズを検索する一般化された手法（３４００）を示す。したがって、デコーダは、オーディオシーケンス全体または他のレベルについて、タイルのチャネルごと、タイルごと、またはフレームごとにマスク要素の量子化ステップサイズを変更することができる。

【０２８６】

デコーダは、１つまたは複数のマスクの量子化ステップサイズを入手する（３４１０）ことによって開始する（影響されるマスクの数は、エンコーダが柔軟な量子化ステップサイズを割り当てるレベルに依存する）。一実施形態では、量子化ステップサイズが、マスク情報の１ｄＢ、２ｄＢ、３ｄＢ、または４ｄＢである。代替案では、エンコーダおよびデコーダが、マスク情報の異なるまたは追加の量子化ステップサイズを使用する。

【０２８７】
次に、デコーダは、マスク情報の量子化ステップサイズを使用して１つまたは複数の量子化行列を逆量子化し（３４２０）、マルチチャネルオーディオデータを再構成する。
【０２８８】
２．量子化行列の時間予測
図３５に、時間予測を使用して量子化行列を圧縮する一般化された手法（３５００）を示す。手法（３５００）では、エンコーダが、マスク値の時間相関を活用する。これによって量子化行列に関連するビットレートが下がる。
【０２８９】
図３５および３６に、オーディオデータのフレームのチャネルでの量子化行列の時間予測を示す。代替案では、エンコーダが、複数のフレームの間、オーディオの他のシーケンスで、または量子化行列の異なる構成で、時間予測を使用して量子化行列を圧縮する。
【０２９０】
図３５を参照すると、エンコーダは、フレームの量子化行列を入手する（３５１０）。チャネルの量子化行列は、ウィンドウからウィンドウへと同一のままである傾向があり、予測コーディングのよい候補になる。
【０２９１】
エンコーダは、時間予測を使用して量子化行列をエンコードする（３５２０）。たとえば、エンコーダは、図３６に示された手法（３６００）を使用する。代替案では、エンコーダは、時間予測を用いる別の手法を使用する。
【０２９２】
エンコーダは、圧縮する行列がまだあるかどうかを判定し（３５３０）、そうでない場合には終了する。それ以外の場合には、エンコーダは、次の量子化行列を入手する。たとえば、エンコーダは、次のフレームの行列がエンコードに使用可能であるかどうかを検査する。
【０２９３】
図３６に、一実施形態で時間予測を使用してチャネルの量子化行列を圧縮するより詳細な手法（３６００）を示す。時間圧縮では、異なるウィンドウサイズのタイルにまたがる再サンプリング処理を使用し、予測残差に対するランレベルコーディングを使用して、ビットレートを下げる。
【０２９４】
エンコーダは、次に圧縮される量子化行列の圧縮を開始し（３６１０）、アンカ行列が使用可能であるかどうかを検査する（３６２０）が、これは、通常は、行列がそのチャネルの最初の行列であるかどうかに依存する。アンカ行列が使用可能でない場合には、エンコーダは、量子化行列を直接に圧縮する（３６３０）。たとえば、エンコーダは、量子化行列の要素を差分エンコードし（要素の差分は前の帯域の要素に対するものである）、ハフマンコードを差分に割り当てる。行列の最初の要素（すなわち、帯域０のマスク要素）について、エンコーダは、マスク要素の量子化ステップサイズに依存する予測定数を使用する。

$$PredConst = 45 / MaskQuantMultiplier_{iChannel} \quad (19)$$

代替案では、エンコーダが、アンカ行列の別の圧縮手法を使用する。
【０２９５】
エンコーダは、フレームのチャネルのアンカ行列として量子化行列をセットする（３６４０）。エンコーダがタイルを使用する時には、チャネルのアンカ行列を含むタイルを、アンカタイルと呼ぶことができる。エンコーダは、アンカ行列サイズまたはアンカタイルのタイルサイズを記録するが、これは、異なるサイズを有する行列の予測を形成するのに使用することができる。
【０２９６】
その一方で、アンカ行列が使用可能である場合には、エンコーダは、時間予測を使用し

て量子化行列を圧縮する。エンコーダは、チャネルのアンカ行列に基づいて、量子化行列の予測を計算する（３６５０）。圧縮される量子化行列が、アンカ行列と同一の数の帯域を有する場合には、予測は、アンカ行列の要素である。しかし、圧縮される量子化行列が、アンカ行列と異なる数の帯域を有する場合には、エンコーダは、アンカ行列を再サンプリングして、予測を計算する。

【０２９７】

再サンプリング処理では、圧縮される量子化行列のサイズ／現在のタイルサイズと、アンカ行列のサイズ／アンカタイルサイズを使用する。

$$MaskPrediction[iBand]=AnchorMask[iScaledBand] \quad (20)$$

ここで、ｉＳｃａｌｅｄＢａｎｄは、ｉＢａｎｄの代表的な（たとえば平均）周波数を含むアンカ行列帯域である。ｉＢａｎｄは、現在の量子化行列／現在のタイルサイズの項であり、ｉＳｃａｌｅｄＢａｎｄは、アンカ行列／アンカタイルサイズの項である。

【０２９８】

図３７に、エンコーダがタイルを使用する時の、アンカ行列の再サンプリングの１つの手法を示す。図３７には、予測を形成するための、アンカタイルの帯域への現在のタイルの帯域の例の写像（３７００）が示されている。現在のタイルの量子化行列の帯域境界の中央の周波数（３７２０）が、アンカタイルのアンカ行列の周波数に写像（３７３０）される。マスク予測の値は、写像された周波数が、アンカタイルのアンカ行列の帯域境界（３７１０）に関してどこにあるかに依存してセットされる。代替案では、エンコーダが、チャネル内の前の量子化行列または他の前の行列に関する時間予測を使用するか、別の再サンプリング手法を使用する。

【０２９９】

図３６に戻って、エンコーダは、予測に関する量子化行列の残差を計算する（３６６０）。理想的には、予測が、完全であり、残差が、エネルギを有しない。しかし、必要な場合に、エンコーダは、残差をエンコードする（３６７０）。たとえば、エンコーダは、予測残差について、ランレベルコーディングまたは別の圧縮手法を使用する。

【０３００】

次に、エンコーダは、圧縮される行列がまだあるかどうかを判定し（３６８０）、そうでない場合に、終了する。それ以外の場合に、エンコーダは、次の量子化行列を入手し（３６１０）、継続する。

【０３０１】

図３８に、特定のビットストリーム構文による時間予測を使用して圧縮された量子化行列の検索およびデコーディングの手法（３８００）を示す。量子化行列は、フレームの単一のタイルのチャネルに関するものである。図３８に、ビットストリームの情報を解析するためにデコーダによって実行される手法（３８００）を示し、エンコーダは、対応する手法を実行する。代替案では、デコーダおよびエンコーダが、図３８に示されたオプションの１つまたは複数について別の構文、たとえば、異なるフラグまたは異なる順序付けを使用する構文、またはタイルを使用しない構文を使用する。

【０３０２】

デコーダは、エンコーダがフレームの初めに達したかどうかを検査する（３８１０）。そうである場合には、デコーダは、そのフレームのすべてのアンカ行列に、セットされていないものとしてマークをつける（３８１２）。

【０３０３】

次に、デコーダは、アンカ行列が、次にエンコードされる量子化行列のチャネルで使用可能であるかどうかを検査する（３８２０）。アンカ行列が使用可能でない場合には、デコーダは、チャネルの量子化行列の量子化ステップサイズを入手する（３８３０）。一実施形態では、デコーダが、１ｄＢ、２ｄＢ、３ｄＢ、または４ｄＢの値を入手する。

$$MaskQuantMultiplier_{iChannel}=getBits(2)+1 \quad (21)$$

【０３０４】
デコーダは、チャネルのアンカ行列をデコードする（３８３２）。たとえば、デコーダは、アンカ行列の差分コーディングされた要素をハフマンデコードし（要素の差分が、前の帯域の要素に対するものである）、要素を再構成する。最初の要素について、デコーダは、エンコーダで使用された予測定数を使用する。

$$PredConst=45/MaskQuantMultiplier_{iChannel}\quad(22)$$

代替案では、デコーダが、フレームのチャネルのアンカ行列に別の圧縮解除手法を使用する。

【０３０５】
デコーダは、フレームのチャネルのアンカ行列として量子化行列をセットし（３８３４）、チャネルの量子化行列の値に、アンカ行列の値をセットする。

$$Q_{m,iChannel,iBand}=AnchorMask[iBand]\quad(23)$$

【０３０６】
デコーダは、アンカタイルのタイルサイズも記録するが、これは、アンカタイルと異なるサイズを有するタイルの行列の予測を形成するのに使用することができる。

【０３０７】
その一方で、アンカ行列がチャネルについて使用可能である場合には、デコーダは、時間予測を使用して量子化行列を圧縮解除する。デコーダは、チャネルのアンカ行列に基づいて、量子化行列の予測を計算する（３８４０）。現在のタイルの量子化行列が、アンカ行列と同一の数の帯域を有する場合には、予測は、アンカ行列の要素である。しかし、現在のタイルの量子化行列が、アンカ行列と異なる数の帯域を有する場合には、エンコーダは、アンカ行列を再サンプリングして、たとえば図３７に示された現在のタイルサイズおよびアンカタイルサイズを使用して、予測を入手する。

$$MaskPrediction[iBand]=AnchorMask[iScaledBand]\quad(24)$$

【０３０８】
代替案では、デコーダが、そのチャネルの前の量子化行列または他の前の行列に対する相対的な時間予測を使用するか、別の再サンプリング手法を使用する。

【０３０９】
デコーダは、ビットストリームの次のビットを入手し（３８４２）、ビットストリームに量子化行列の残差が含まれるかどうかを検査する（３８５０）。現在のタイルのこのチャネルに関するマスク更新がない場合には、マスク予測残差が０であり、したがって、

$$Q_{m,iChannel,iBand}=MaskPrediction[iBand]\quad(25)$$

になる。

【０３１０】
その一方で、予測残差がある場合には、デコーダは、たとえばランレベルデコーディングまたは他の圧縮解除手法を使用して、残差をデコードする（３８５２）。次に、デコーダは、予測に予測残差を加算して（３８５４）、量子化行列を再構成する。たとえば、加算は、現在のチャネルｉＣｈａｎｎｅｌの帯域ｉＢａｎｄに関する要素を入手するために、帯域ごとの単純なスカラ加算である。

$$Q_{m,iChannel,iBand}=MaskPrediction[iBand]+MaskPredResidual[iBand]\quad(26)$$

【０３１１】
その後、デコーダは、現在のタイルのすべてのチャネルの量子化行列がデコードされたかどうかを検査し（３８６０）、そうである場合には終了する。そうでない場合には、デコーダは、現在のタイルの次の量子化行列のデコードを継続する。

【０３１２】
Ｄ．組み合わされた逆量子化よび逆重みづけ

デコーダは、必要な量子化および重みづけの情報のすべてを検索したならば、オーディオデータを逆量子化し、逆重みづけする。一実施形態では、デコーダが、逆量子化および逆重みづけを１ステップで実行するが、これを、印刷を明瞭にするために下の２つの式に示す。

【０３１３】

【数１５】

$$CombinedQ = Q_t + Q_{c,iChannel} - \left(Max\left(Q_{m,iChannel,*}\right) - Q_{m,iChannel,iBand}\right) \cdot MaskQuantMultiplier_{iChannel} \quad (27a)$$

$$y_{iqw}[n] = 10^{CombinedQ/20} \cdot x_{iqw}[n] \quad (27b)$$

【０３１４】

ここで、x_{iqw}は、チャネルｉＣｈａｎｎｅｌの入力（たとえば、逆マルチチャネル変換された係数）であり、ｎは、帯域ｉＢａｎｄの係数インデックスである。Ｍａｘ（$Q_{m,iChannel,*}$）は、すべての帯域にわたるチャネルｉＣｈａｎｎｅｌの最大マスク値である（マスクの最大重みづけ係数と最小重みづけ係数の間の差は、通常は、マスク要素の潜在的な値の範囲よりはるかに小さく、したがって、重みづけ係数ごとの量子化調整の量は、最大値に対して相対的に計算される）。$MaskQuantMultiplier_{iChannel}$は、チャネルｉＣｈａｎｎｅｌの量子化行列のマスク量子化ステップ乗数であり、y_{iqw}は、このステップの出力である。

【０３１５】

代替案では、デコーダが、逆量子化および重みづけを、別々にまたは異なる手法を使用して、実行する。

【０３１６】

ＶＩＩ．マルチチャネル後処理

いくつかの実施形態で、図７のデコーダ（７００）などのデコーダが、時間領域の再構成されるオーディオサンプルに対するマルチチャネル後処理を実行する。

【０３１７】

マルチチャネル後処理は、多数の異なる目的に使用することができる。たとえば、デコードされるチャネルの数が、出力のチャネル数より少ない場合がある（たとえば、コーディングの複雑さまたはバッファ満杯度を下げるために、エンコーダが１つまたは複数の入力チャネルまたはマルチチャネル変換されたチャネルを捨てたので）。その場合に、マルチチャネル後処理変換を使用して、デコードされたチャネルの実際のデータに基づいて、１つまたは複数のファントムチャネルを作成することができる。あるいは、デコードされるチャネルの数が出力チャネルの数と等しい場合であっても、提示の任意の空間回転、スピーカ位置の間での出力チャネルの再写像、または他の立体感あるいは特殊効果に後処理変換を使用することができる。あるいは、デコードされるチャネルの数が出力チャネルの数より多い（たとえば、サラウンドサウンドオーディオをステレオ機器で再生する時）場合に、後処理変換を使用して、チャネルを「折り畳む」ことができる。いくつかの実施形態で、折り曲げられた係数が、潜在的に経時的に変化し、マルチチャネル後処理が、ビットストリームによって制御される。これらのシナリオおよび応用例の変換行列を、エンコーダによって提供またはシグナリングすることができる。

【０３１８】

図３９に、マルチチャネル後処理の一般化された手法（３９００）を示す。デコーダは、図７に示された手法または他の圧縮解除手法を使用して、エンコードされたマルチチャネルオーディオデータ（３９０５）をデコードし（３９１０）、再構成された時間領域マルチチャネルオーディオデータ（３９１５）を作る。

【０３１９】
デコーダは、次に、時間領域マルチチャネルオーディオデータ（３９１５）に対してマルチチャネル後処理を実行する（３９２０）。たとえば、エンコーダが、Ｍ個のデコードされるチャネルを作り、デコーダが、Ｎチャネルを出力する時に、後処理に、一般的なＭ対Ｎ変換が含まれる。デコーダは、再構成されるＭ個のコーディングされたチャネルのそれぞれから１つのＭ個の同一位置（時間的に）のサンプルをとり、欠けているチャネル（すなわちエンコーダによって捨てられたＮ－Ｍ個のチャネル）を、０でパディングする。デコーダは、Ｎ個のサンプルに行列A_{post}をかける。

$$y_{post} = A_{post} \cdot x_{post} \qquad (28)$$

ここで、x_{post}およびy_{post}は、マルチチャネル後処理へのＮチャネルの入力および出力であり、A_{post}は、一般的なＮ×Ｎ変換行列であり、x_{post}は、出力ベクトル長Ｎに一致するように０でパディングされる。

【０３２０】
行列A_{post}は、事前に決定された要素を有する行列とすることができ、あるいは、エンコーダによって指定される要素を有する一般的な行列とすることができる。エンコーダは、事前に決定された行列を使用するようにデコーダに知らせる（たとえば、１つまたは複数のフラグビットを用いて）か、一般的な行列の要素をデコーダに送ることができ、あるいは、同一の行列A_{post}を必ず使用するようにデコーダを構成することができる。行列A_{post}は、対象または可逆など、特殊な特性を有する必要はない。追加の柔軟性のために、マルチチャネル後処理を、フレームごとまたは他の基礎でオン／オフにすることができる（この場合に、デコーダは、単位行列を使用して、チャネルを未変更のままにすることができる）。

【０３２１】
図４０に、図４に示された順序のチャネルを有する５．１チャネル再生環境で左チャネルおよび右チャネルからファントム中央チャネルを作成するのに使用される例の行列$A_{p-center}$（４０００）を示す。例の行列$A_{p-center}$（４０００）は、他のチャネルを変更せずに渡す。デコーダは、左、右、サブウーファ、左後ろ、および右後ろのチャネルから時間的に同位置のサンプルを入手し、中央チャネルを０でパディングする。その後、デコーダは、６つの入力サンプルに行列$A_{p-center}$（４０００）をかける。

【０３２２】
【数１６】

$$\begin{bmatrix} a \\ b \\ \frac{a+b}{2} \\ d \\ e \\ f \end{bmatrix} = A_{P-Center} \cdot \begin{bmatrix} a \\ b \\ 0 \\ d \\ e \\ f \end{bmatrix} \qquad (29)$$

【０３２３】
代替案では、デコーダが、異なる係数を有する行列または異なる個数のチャネルを使用する。たとえば、デコーダは、行列を使用して、５．１マルチチャネルオーディオのコーディングされたチャネルから、７．１チャネル、９．１チャネル、または異なる再生環境でのファントムチャネルを作成する。

【０３２４】
図４１に、フレームごとに変換行列が潜在的に変化するマルチチャネル後処理の手法（４１００）を示す。変換行列の変更は、注意深く扱われない場合に、最終的な出力の可聴雑音（たとえばポンという音）につながる可能性がある。ポンという雑音を導入しないようにするために、デコーダは、ある変換行列から別の変換行列へ、フレームの間に徐々に

推移する。

【０３２５】

デコーダは、まず、図７に示された手法または他の圧縮解除手法を使用して、フレームのエンコードされたマルチチャネルオーディオデータをデコードし（４１１０）、再構成された時間領域マルチチャネルオーディオデータを作る。次に、デコーダは、たとえば図４２に示されているように、フレームの後処理行列を入手する（４１２０）。

【０３２６】

デコーダは、（前のフレームがある場合に）現在のフレームの行列が前のフレームの行列と異なるかどうかを判定する（４１３０）。現在の行列が同一であるか、前の行列が存在しない場合には、デコーダは、現在のフレームの再構成されたオーディオサンプルに行列を適用する（４１４０）。そうでない場合には、デコーダは、現在のフレームの再構成されたオーディオサンプルにブレンドされた変換行列を適用する（４１５０）。ブレンディング関数は、実施形態に依存する。一実施形態では、現在のフレームのサンプルｉで、デコーダが、短期間ブレンドされた行列$A_{post,i}$を使用する。

【０３２７】

【数１７】

$$A_{post,i} = \frac{NumSamples - i}{NumSamples} A_{post,prev} + \frac{i}{NumSamples} A_{post,current} \qquad (30)$$

【０３２８】

ここで、$A_{post,prev}$および$A_{post,current}$は、前のフレームおよび現在のフレームの後処理行列であり、ＮｕｍＳａｍｐｌｅｓは、現在のフレームのサンプル数である。代替案では、デコーダが、別のブレンディング関数を使用して、後処理変換行列の不連続性を平滑化する。

【０３２９】

デコーダは、フレームごとに手法（４１００）を繰り返す。代替案では、デコーダは、他の基礎でマルチチャネル後処理を変更する。

【０３３０】

図４２に、特定のビットストリーム構文によるマルチチャネル後処理の変換行列を識別し、検索する手法（４２００）を示す。この構文を用いると、事前定義された変換行列ならびにマルチチャネル後処理のカスタム行列の指定が可能になる。図４２には、ビットストリームを解析するためにデコーダによって実行される手法（４２００）が示され、エンコーダは、対応する手法（フラグの設定、要素のデータのパックなど）を実行して、ビットストリーム構文に従って変換行列をフォーマットする。代替案では、デコーダおよびエンコーダが、図４２に示されたオプションの１つまたは複数に、別の構文、たとえば、異なるフラグまたは異なる順序付けを使用する構文を使用する。

【０３３１】

まず、デコーダは、チャネルの数＃Ｃｈａｎｎｅｌｓが１より大きいかどうかを判定する（４２１０）。＃Ｃｈａｎｎｅｌｓが１である場合には、オーディオデータはモノラルであり、デコーダは、単位行列を使用する（４２１２）（すなわち、マルチチャネル後処理自体を実行しない）。

【０３３２】

その一方で、＃Ｃｈａｎｎｅｌｓ＞１の場合には、デコーダは、ビットストリームの次のビットと等しくなるように一時値ｉＴｍｐをセットする（４２２０）。次に、デコーダは、一時値の値を検査するが（４２３０）、この値によって、デコーダが単位行列を使用しなければならない（４２３２）か否かが示される。

【０３３３】

デコーダが、マルチチャネルオーディオに単位行列以外の何かを使用する場合には、デコーダは、ビットストリームの次のビットと等しくなるように一時値ｉＴｍｐをセットする（４２４０）。次に、デコーダは、一時値の値を検査するが（４２５０）、この値によ

って、デコーダが事前定義のマルチチャネル変換行列を使用（４２５２）しなければならないか否かが示される。デコーダが、事前定義の行列を使用する（４２５２）場合には、デコーダは、複数の使用可能な事前定義の行列のどれをデコーダが使用しなければならないかを示す１つまたは複数の追加ビットをビットストリーム（図示せず）から入手することができる。

【０３３４】

デコーダが、事前定義の行列を使用しない場合には、デコーダは、カスタム行列をデコードするために、さまざまな一時値を初期化する。デコーダは、終了した係数のカウンタｉＣｏｅｆｓＤｏｎｅに０をセットし（４２６０）、行列の要素数（＃Ｃｈａｎｎｅｌｓ^2）と等しくなるように、デコードする係数の数＃ＣｏｅｆｓＴｏＤｏをセットする（４２６２）。特定の特性（たとえば対照）を有することが既知の行列について、デコードされる係数の数を減らすことができる。次に、デコーダは、すべての係数がビットストリームから検索されたかどうかを判定し（４２７０）、そうである場合には終了する。そうでない場合には、デコーダは、行列の次の要素の値Ａ［ｉＣｏｅｆｓＤｏｎｅ］を入手し（４２７２）、ｉＣｏｅｆｓＤｏｎｅを増分する（４２７４）。要素がコーディングされ、ビットストリームにパックされる形は、実装依存である。図４２では、構文によって、変換行列の要素ごとに４ビットの精度が可能であり、各要素の絶対値が、１以下である。他の実施形態では、要素ごとの精度が、異なり、エンコーダおよびデコーダが、変換行列の冗長性のパターンを活用する圧縮を使用し、かつ／または構文が、他の形で異なる。

【０３３５】

好ましい実施形態に関して本発明の原理を説明し、示したが、説明された実施形態を、そのような原理から逸脱せずに、配置および詳細において修正できることを諒解されたい。本明細書に記載のプログラム、処理、または方法は、特に示されない限り、コンピューティング環境の特定のタイプに関係せず、制限されないことを理解されたい。さまざまなタイプの汎用コンピューティング環境および特殊化されたコンピューティング環境は、本明細書に記載の教示による動作と共に使用されるか、その動作を実行することができる。説明された実施形態の、ソフトウェアで示された要素は、ハードウェアで実施することができ、逆も同様である。

【０３３６】

本発明の原理を適用できる多数の可能な実施形態に鑑みて、本発明者は、そのような実施形態のすべてを、請求項およびその均等物の範囲および趣旨に含めることができると主張する。

【図面の簡単な説明】

【０３３７】

【図１】従来技術によるオーディオエンコーダのブロック図である。

【図２】従来技術によるオーディオデコーダのブロック図である。

【図３ａ】従来技術によるステレオオーディオデータのフレームのウィンドウ構成を示す図である。

【図３ｂ】従来技術によるステレオオーディオデータのフレームのウィンドウ構成を示す図である。

【図３ｃ】従来技術によるステレオオーディオデータのフレームのウィンドウ構成を示す図である。

【図４】５．１チャネル／スピーカ構成の６つのチャネルを示す図である。

【図５】説明されている本実施形態を実施することができる適切なコンピューティング環境のブロック図である。

【図６】説明されている本実施形態を実施することができるオーディオエンコーダのブロック図である。

【図７】説明されている本実施形態を実施することができるオーディオデコーダのブロック図である。

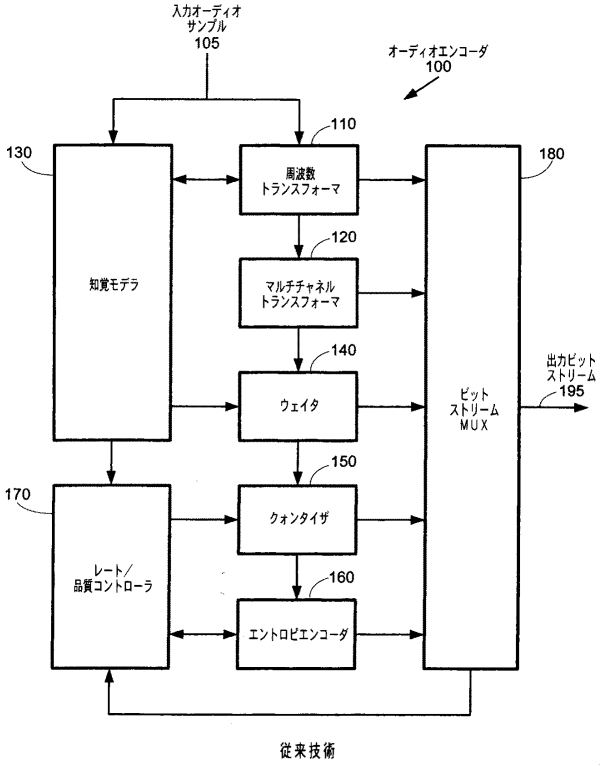
【図８】マルチチャネル前処理の一般化された手法を示す流れ図である。

【図９ａ】マルチチャネル前処理の例の行列を示す図である。
【図９ｂ】マルチチャネル前処理の例の行列を示す図である。
【図９ｃ】マルチチャネル前処理の例の行列を示す図である。
【図９ｄ】マルチチャネル前処理の例の行列を示す図である。
【図９ｅ】マルチチャネル前処理の例の行列を示す図である。
【図１０】フレームごとに変換行列が潜在的に変化するマルチチャネル前処理の手法を示す流れ図である。
【図１１ａ】マルチチャネルオーディオの例のタイル構成を示す図である。
【図１１ｂ】マルチチャネルオーディオの例のタイル構成を示す図である。
【図１２】マルチチャネルオーディオのタイルを構成する一般化された手法を示す流れ図である。
【図１３】特定のビットストリーム構文によるマルチチャネルオーディオの並列のタイル構成およびタイル情報送出の手法を示す流れ図である。
【図１４】知覚的重みづけの後にマルチチャネル変換を実行する一般化された手法を示す流れ図である。
【図１５】逆知覚的重みづけの前に逆マルチチャネル変換を実行する一般化された手法を示す流れ図である。
【図１６】一実施形態でマルチチャネル変換についてタイル内のチャネルをグループ化する手法を示す流れ図である。
【図１７】特定のビットストリーム構文によるビットストリームからのタイルのチャネルグループ情報およびマルチチャネル変換情報の検索の手法を示す流れ図である。
【図１８】一実施形態でマルチチャネル変換にチャネルグループの周波数帯域を選択的に含める手法を示す流れ図である。
【図１９】特定のビットストリーム構文によるビットストリームからのタイルのチャネルグループに関するマルチチャネル変換の帯域オン／オフ情報を検索する手法を示す流れ図である。
【図２０】より単純なマルチチャネル変換の階層を使用してマルチチャネル変換をエミュレートする一般化された手法を示す流れ図である。
【図２１】マルチチャネル変換の例の階層を示す図である。
【図２２】特定のビットストリーム構文によるビットストリームからのチャネルグループに関するマルチチャネル変換の階層の情報を検索する手法を示す流れ図である。
【図２３】複数の使用可能なタイプの中からマルチチャネル変換タイプを選択する一般化された手法を示す流れ図である。
【図２４】複数の使用可能なタイプの中からマルチチャネル変換タイプを検索し、逆マルチチャネル変換を実行する手法を示す流れ図である。
【図２５】特定のビットストリーム構文によるビットストリームからチャネルグループに関するマルチチャネル変換情報を検索する手法を示す流れ図である。
【図２６】マルチチャネル変換行列を表すギブンス回転の回転行列の一般形を示す図である。
【図２７ａ】マルチチャネル変換行列を表すギブンス回転の例の回転行列を示す図である。
【図２７ｂ】マルチチャネル変換行列を表すギブンス回転の例の回転行列を示す図である。
【図２７ｃ】マルチチャネル変換行列を表すギブンス回転の例の回転行列を示す図である。
【図２８】量子化されたギブンス因数分解回転を使用してマルチチャネル変換行列を表す一般化された手法を示す流れ図である。
【図２９】特定のビットストリーム構文によるビットストリームからチャネルグループの汎用ユニタリ変換の情報を検索する手法を示す流れ図である。
【図３０】特定のビットストリーム構文によるビットストリームからタイルの全体的なタ

イル量子化係数を検索する手法を示す流れ図である。
【図３１】マルチチャネルオーディオデータのチャネルごとの量子化ステップ変更子を計算する一般化された手法を示す流れ図である。
【図３２】特定のビットストリーム構文によるビットストリームからチャネルごとの量子化ステップ変更子を検索する手法を示す流れ図である。
【図３３】量子化行列要素の量子化ステップサイズを適応式にセットする一般化された手法を示す流れ図である。
【図３４】量子化行列要素の適応量子化ステップサイズを検索する一般化された手法を示す流れ図である。
【図３５】時間予測を使用して量子化行列を圧縮する手法を示す流れ図である。
【図３６】時間予測を使用して量子化行列を圧縮する手法を示す流れ図である。
【図３７】量子化行列要素の予測のための帯域の写像を示す図である。
【図３８】特定のビットストリーム構文による時間予測を使用して圧縮された量子化行列の検索およびデコーディングの手法を示す流れ図である。
【図３９】マルチチャネル後処理の一般化された手法を示す流れ図である。
【図４０】マルチチャネル後処理の例の行列を示す図である。
【図４１】フレームごとに変換行列が潜在的に変化するマルチチャネル後処理の手法を示す流れ図である。
【図４２】特定のビットストリームによるマルチチャネル後処理の変換行列を識別し、検索する手法を示す流れ図である。
【符号の説明】
【０３３８】
４００　５．１チャネル／スピーカ配置行列
５００　コンピューティング環境
５１０　処理ユニット
５２０　メモリ
５７０　通信接続
５５０　入力デバイス
５６０　出力デバイス
５４０　ストレージ
６００　オーディオエンコーダ
６０５　入力オーディオサンプル
６０８　セレクタ
６１０　マルチチャネルプリプロセッサ
６２０　パーティショナ／タイルコンフィギュアラ
６３０　周波数トランスフォーマ
６４０　知覚モデラ
６４２　量子化帯域ウェイタ
６４４　チャネルウェイタ
６９０　ＭＵＸ
６９５　出力ビットストリーム
６５０　マルチチャネルトランスフォーマ
６７２　ミックスド／ピュアロスレスコーダ
６７４　エントロピエンコーダ
６８０　レート／品質コントローラ
６６０　クォンタイザ
６７０　エントロピエンコーダ
７００　オーディオデコーダ
７０５　入力ビットストリーム
７１０　ＤＥＭＵＸ

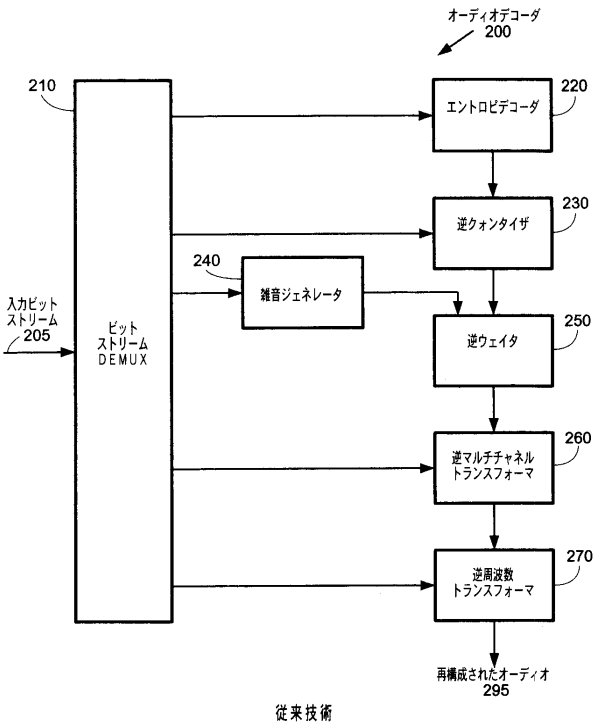
７３０　タイル構成デコーダ
７２０　エントロピデコーダ
７４０　逆マルチチャネルトランスフォーマ
７５０　逆クォンタイザ／ウェイタ
７６０　逆周波数トランスフォーマ
７７０　オーバーラッパ／アダー
７２２　ミックスド／ピュアロスレスデコーダ
７８０　マルチチャネルポストプロセッサ
７９５　再構成されたオーディオ
８０５　時間領域マルチチャネルオーディオデータ
８１５　時間領域マルチチャネル変換されたオーディオデータ
８２５　エンコードされたマルチチャネルオーディオデータ

【図１】



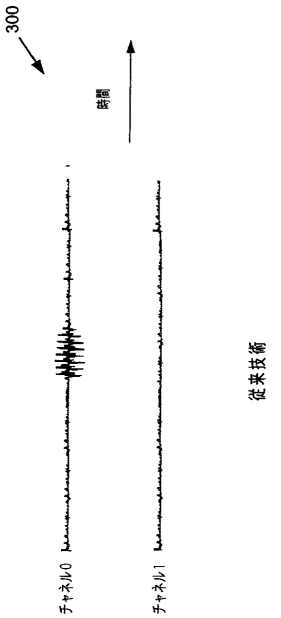
従来技術

【図２】

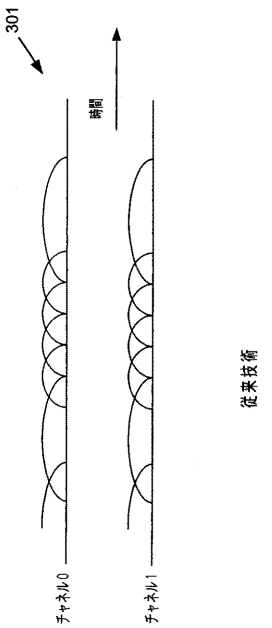


従来技術

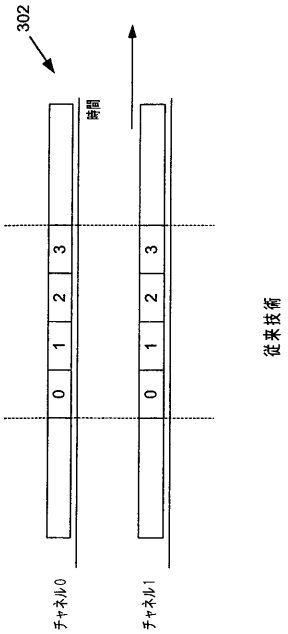
【図３ａ】



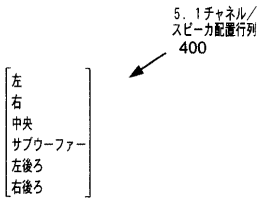
【図３ｂ】



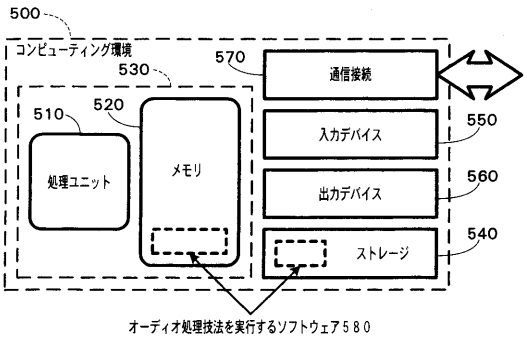
【図３ｃ】



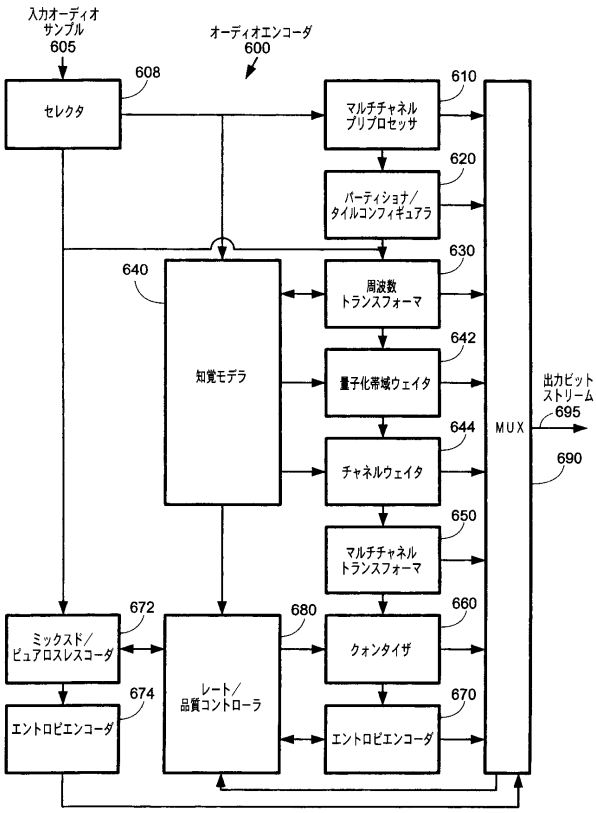
【図４】



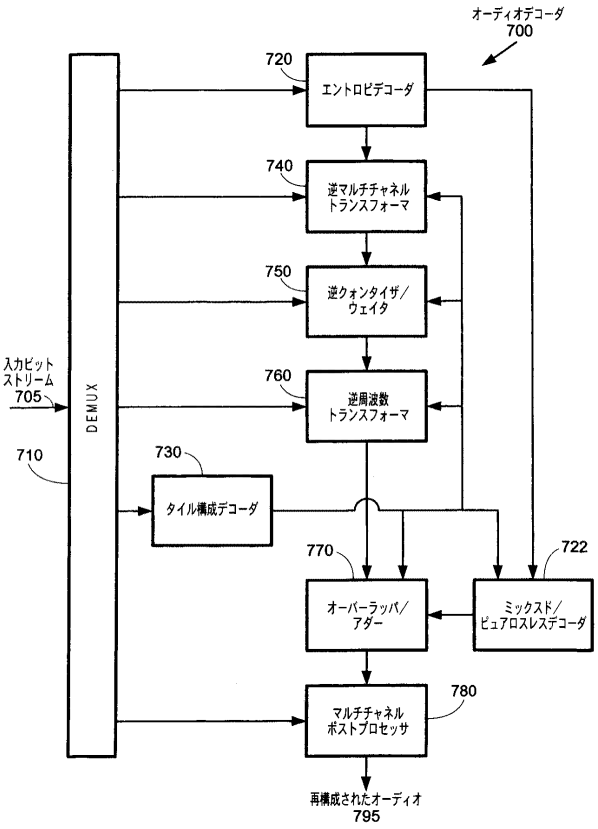
【図５】



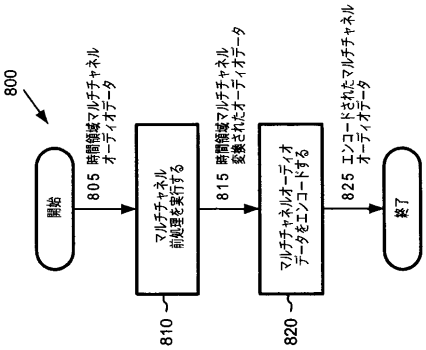
【図６】



【図７】



【図８】



【図９ａ】

$$A_{low} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad 900$$

【図９ｂ】

$$A_{inter,1} = \begin{bmatrix} \left(\frac{1}{1+0.5\cdot\alpha}\right) & 0 & \left(\frac{0.5\cdot\alpha}{1+0.5\cdot\alpha}\right) & 0 & 0 & 0 \\ 0 & \left(\frac{1}{1+0.5\cdot\alpha}\right) & \left(\frac{0.5\cdot\alpha}{1+0.5\cdot\alpha}\right) & 0 & 0 & 0 \\ \left(\frac{\alpha}{1+2\alpha}\right) & \left(\frac{\alpha}{1+2\alpha}\right) & \left(\frac{1}{1+2\alpha}\right) & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \left(\frac{1}{1+\alpha}\right) & \left(\frac{\alpha}{1+\alpha}\right) \\ 0 & 0 & 0 & 0 & \left(\frac{\alpha}{1+\alpha}\right) & \left(\frac{1}{1+\alpha}\right) \end{bmatrix} \quad 901$$

【図９ｃ】

$$A_{high,1} = \begin{bmatrix} \left(\frac{1}{1.5}\right) & 0 & \left(\frac{0.5}{1.5}\right) & 0 & 0 & 0 \\ 0 & \left(\frac{1}{1.5}\right) & \left(\frac{0.5}{1.5}\right) & 0 & 0 & 0 \\ \left(\frac{1}{3}\right) & \left(\frac{1}{3}\right) & \left(\frac{1}{3}\right) & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0.5 \\ 0 & 0 & 0 & 0 & 0.5 & 0.5 \end{bmatrix} \quad 902$$

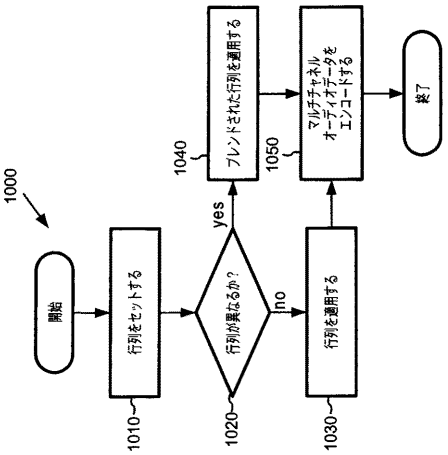
【図９ｄ】

$$A_{inter,2} = \begin{bmatrix} \frac{1}{1+0.5\cdot\alpha} & 0 & \frac{0.5\cdot\alpha}{1+0.5\cdot\alpha} & 0 & 0 & 0 \\ 0 & \frac{1}{1+0.5\cdot\alpha} & \frac{0.5\cdot\alpha}{1+0.5\cdot\alpha} & 0 & 0 & 0 \\ 0.5\cdot\alpha & 0.5\cdot\alpha & 1-\alpha & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{1+\alpha} & \frac{\alpha}{1+\alpha} \\ 0 & 0 & 0 & 0 & \frac{\alpha}{1+\alpha} & \frac{1}{1+\alpha} \end{bmatrix} \quad 903$$

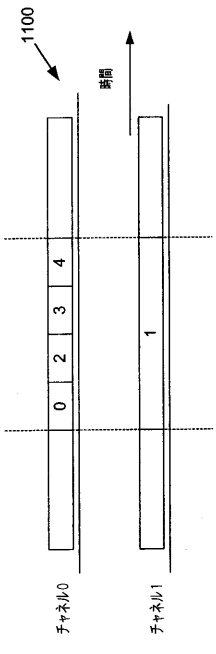
【図９ｅ】

$$A_{high,2} = \begin{bmatrix} \frac{1}{1.5} & 0 & \frac{0.5}{1.5} & 0 & 0 & 0 \\ 0 & \frac{1}{1.5} & \frac{0.5}{1.5} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0.5 \\ 0 & 0 & 0 & 0 & 0.5 & 0.5 \end{bmatrix} \quad 904$$

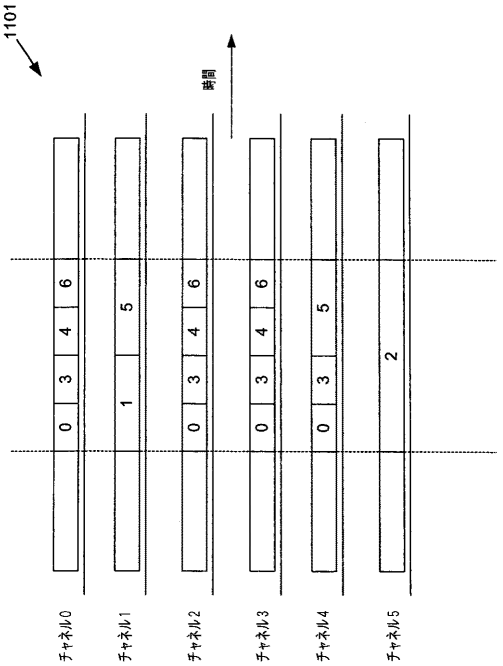
【図１０】



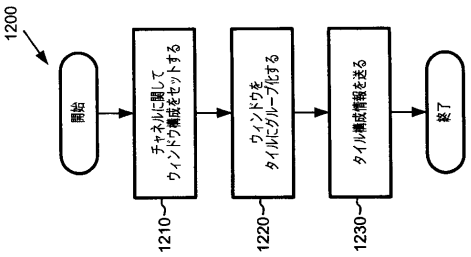
【図１１ａ】



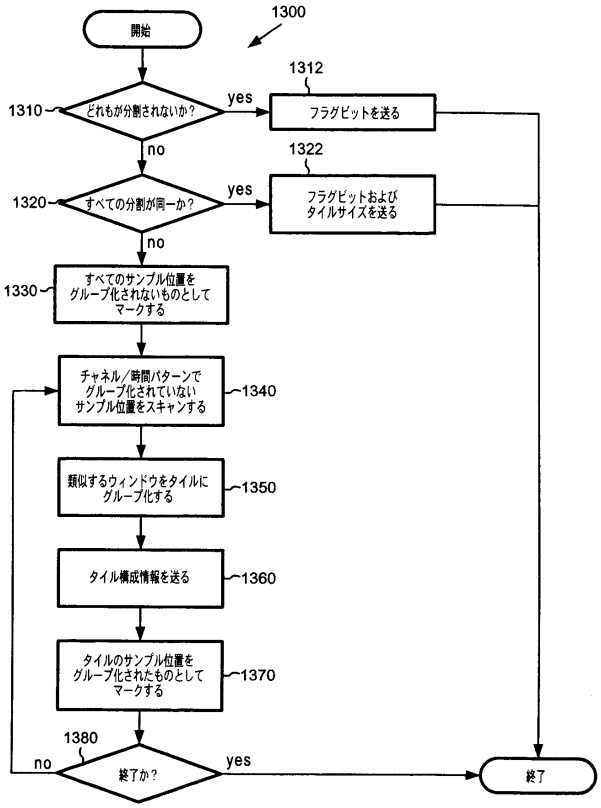
【図１１ｂ】



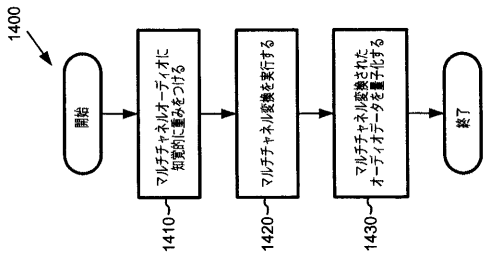
【図１２】



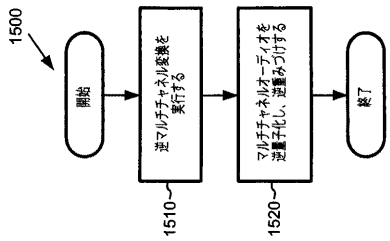
【図１３】



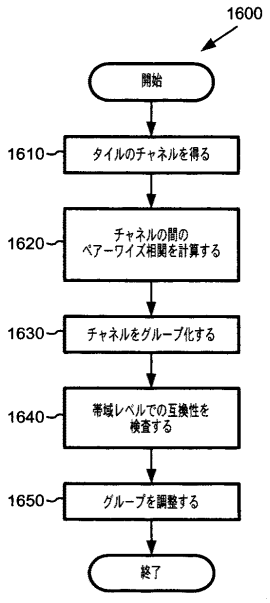
【図１４】



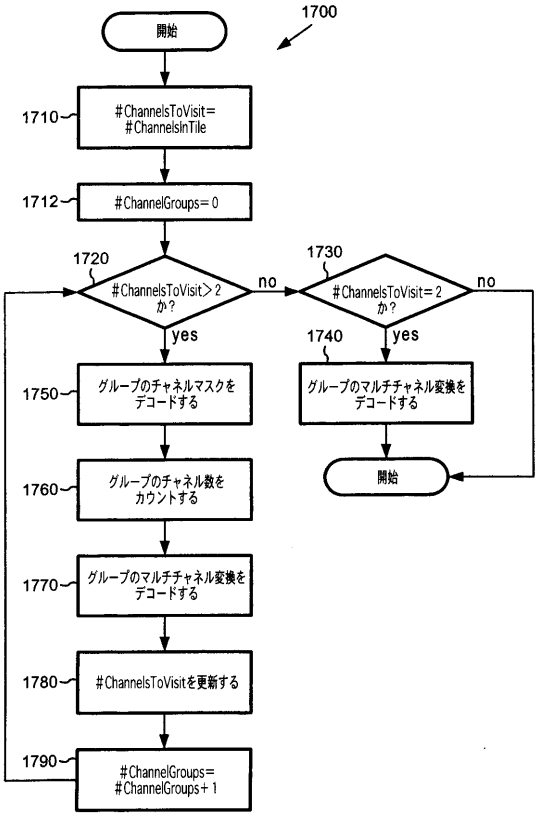
【図１５】



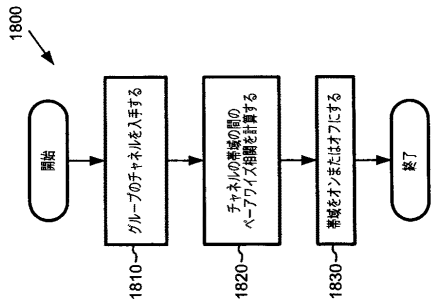
【図１６】



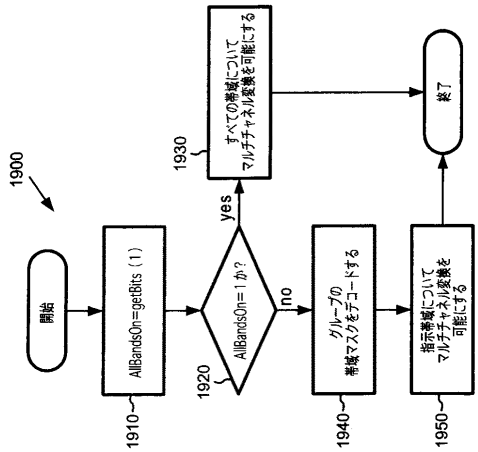
【図１７】



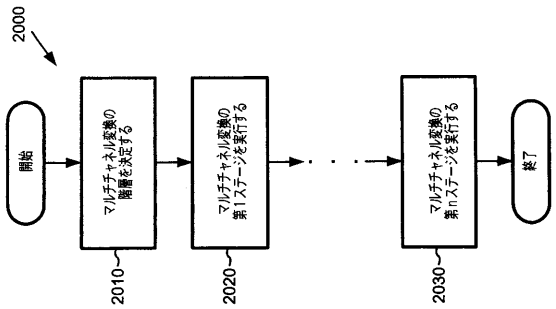
【図１８】



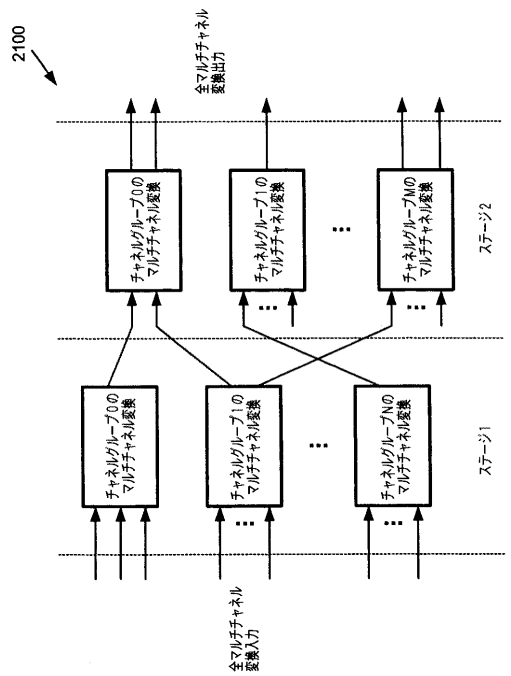
【図１９】



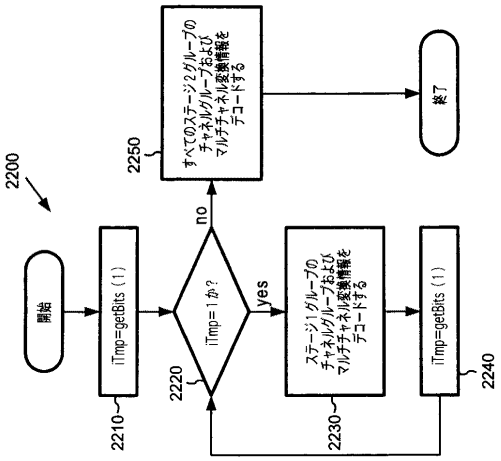
【図２０】



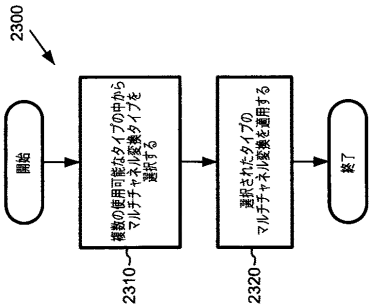
【図２１】



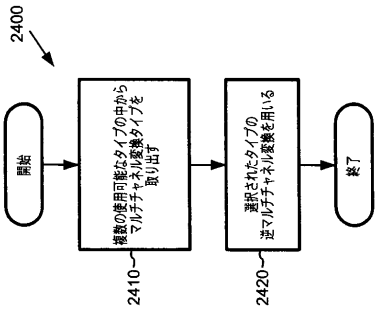
【図２２】



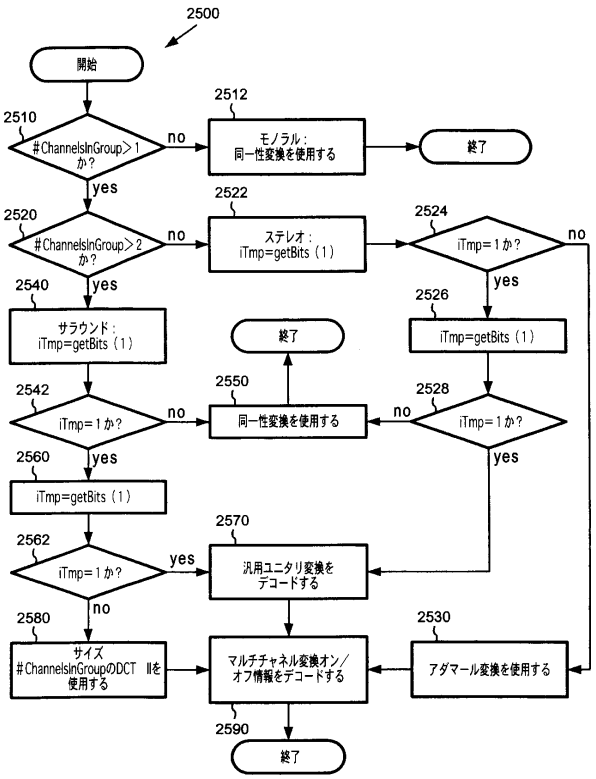
【図２３】



【図２４】



【図２５】



【図２６】

$$\begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & \cos\omega_k & 0 & \cdots & \sin\omega_k & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \cdots & 0 & 0 & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & -\sin\omega_k & 0 & \cdots & \cos\omega_k & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 1 \end{bmatrix}$$

2600

【図２７ａ】

$$\Theta_1 = \begin{bmatrix} \cos\varpi_1 & \sin\varpi_1 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ -\sin\varpi_1 & \cos\varpi_1 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

2700

【図２７ｂ】

$$\Theta_2 = \begin{bmatrix} \cos\varpi_2 & 0 & \sin\varpi_2 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ -\sin\varpi_2 & 0 & \cos\varpi_2 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

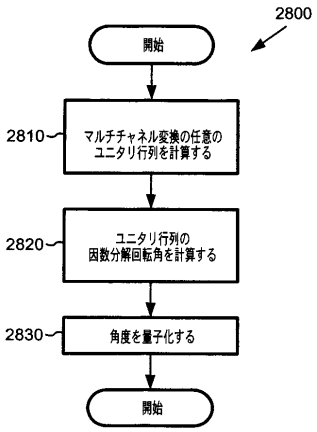
2701

【図２７ｃ】

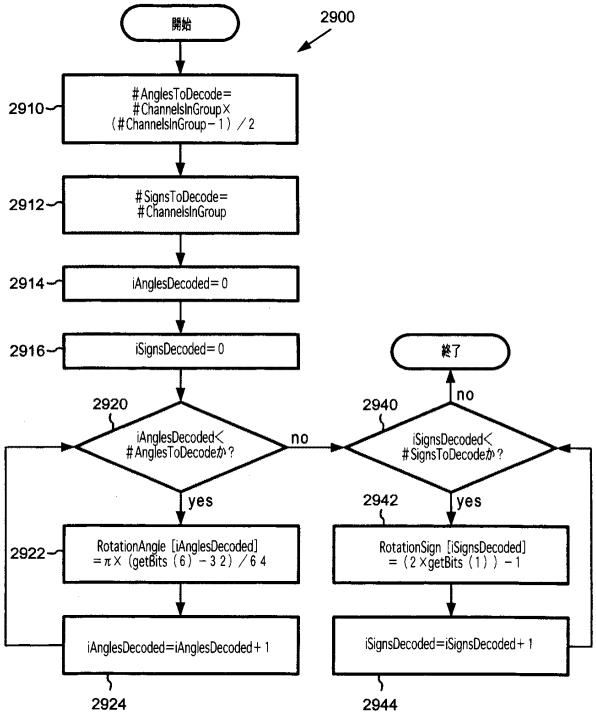
$$\Theta_3 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & \cos\varpi_3 & \sin\varpi_3 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & -\sin\varpi_3 & \cos\varpi_3 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

2702

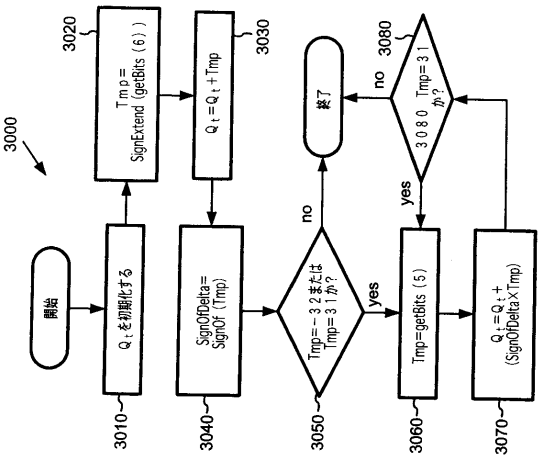
【図２８】



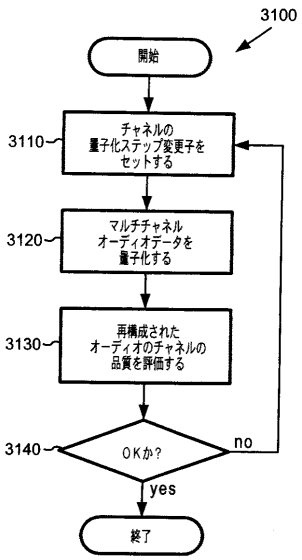
【図２９】



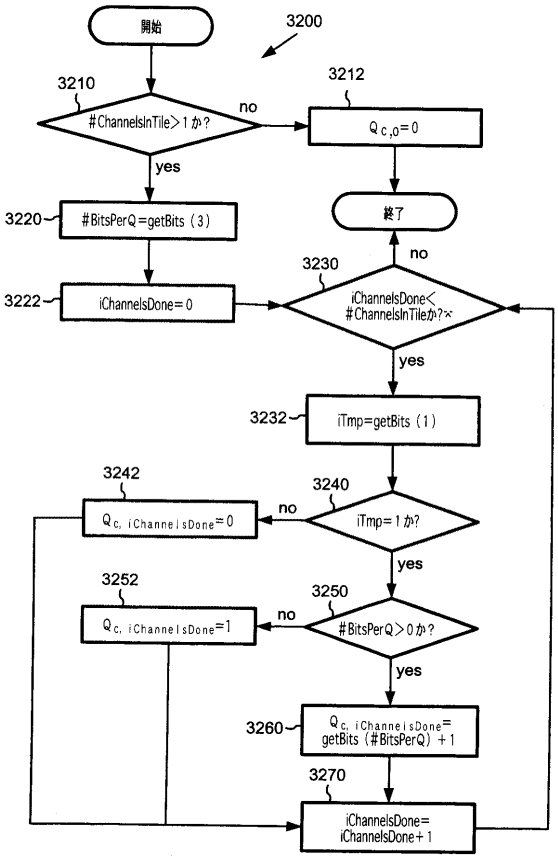
【図３０】



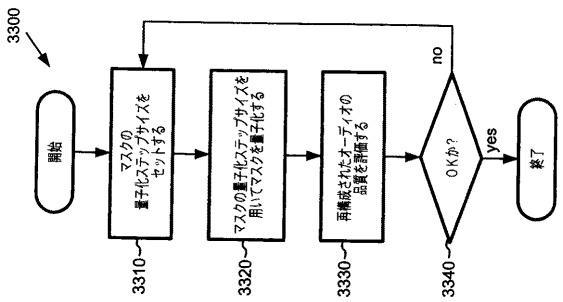
【図３１】



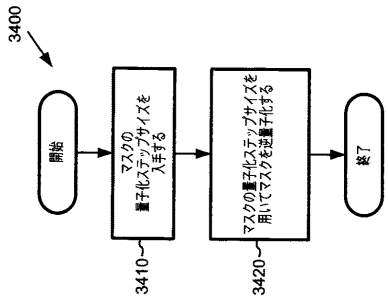
【図３２】



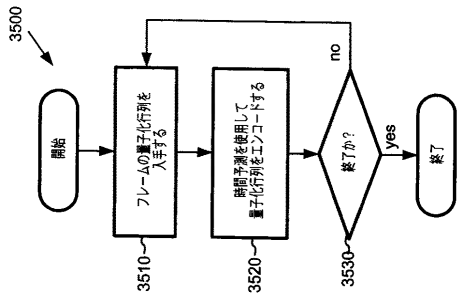
【図３３】



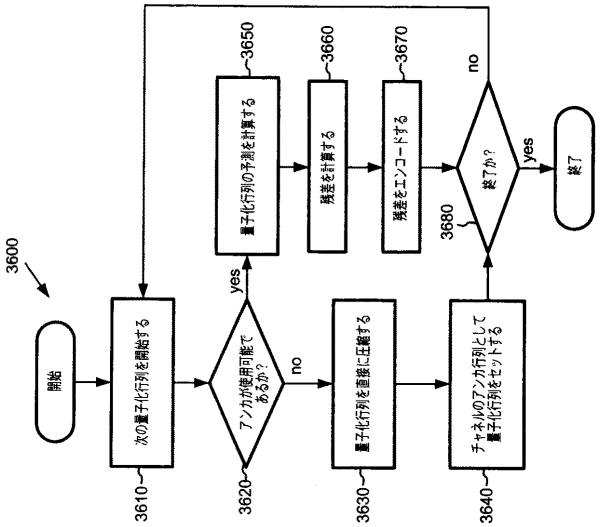
【図３４】



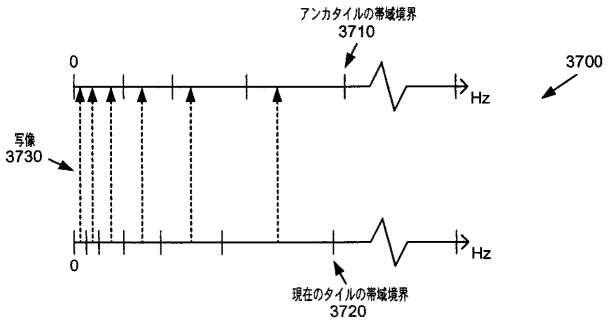
【図３５】



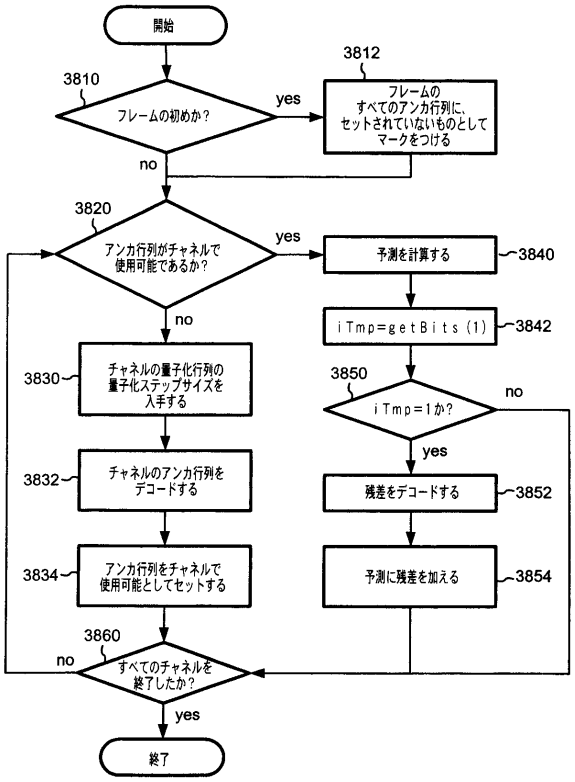
【図３６】



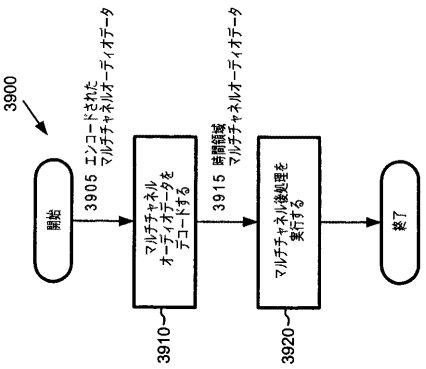
【図３７】



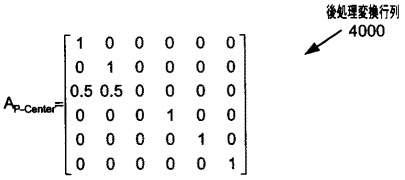
【図３８】



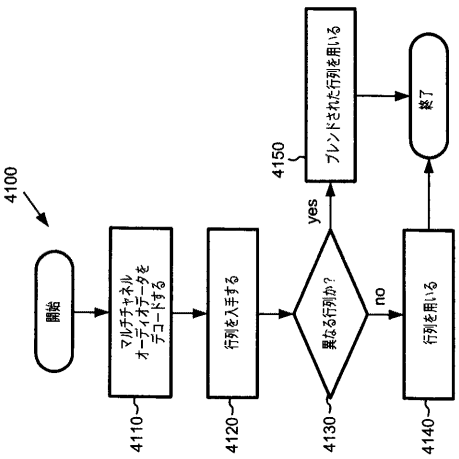
【図３９】



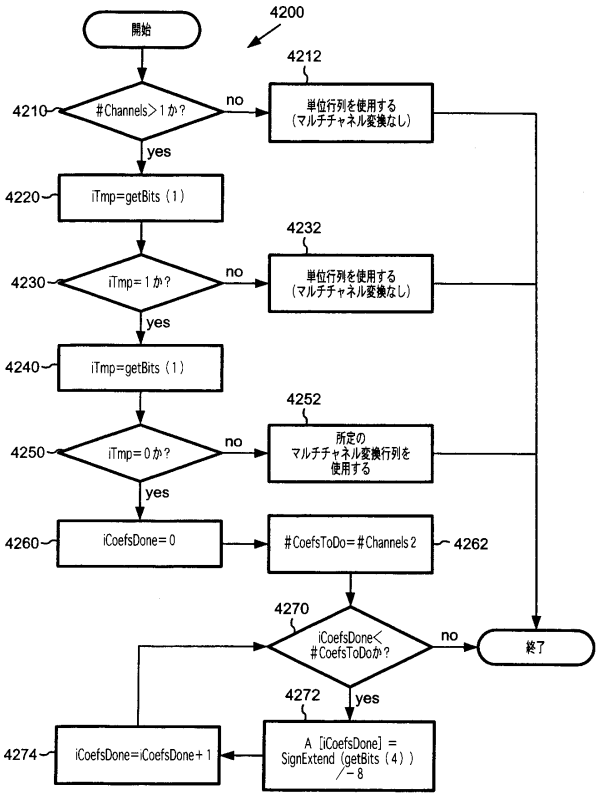
【図４０】



【図４１】



【図４２】



フロントページの続き

(72)発明者　ウェイーゲ　チェン
アメリカ合衆国　９８０２９　ワシントン州　イサコア　サウスイースト　３７　ストリート　２４６３５

審査官　山下　剛史

(56)参考文献　特開２００１－２８５０７３（ＪＰ，Ａ）
特開平６－１４９２９２（ＪＰ，Ａ）
特表２００２－５２０７６０（ＪＰ，Ａ）
特表２００２－５２６７９８（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ１０Ｌ　１９／００－１９／１４
Ｈ０４Ｂ　１４／０４
Ｈ０３Ｍ　７／３０
Ｇ１１Ｂ　２０／１０