(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2007-300389
(P2007-300389A)
(43)公開日　平成19年11月15日(2007.11.15)

| (51)Int.Cl. | | | F I | | | テーマコード（参考） |
|---|---|---|---|---|---|---|
| H04N | 7/32 | (2006.01) | H04N | 7/137 | Z | 5C059 |
| H04N | 7/30 | (2006.01) | H04N | 7/133 | Z | 5J064 |
| H03M | 7/40 | (2006.01) | H03M | 7/40 | | |

審査請求 未請求　請求項の数 4　ＯＬ　（全 19 頁）

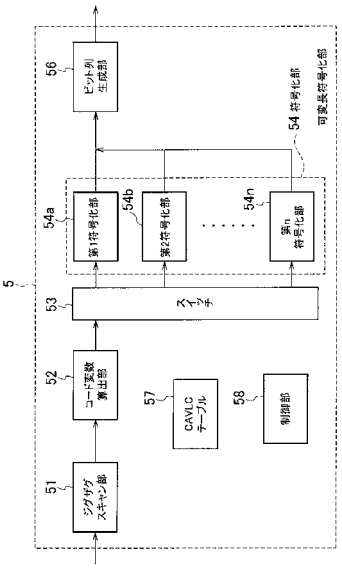
(21)出願番号　特願2006-126585 (P2006-126585)
(22)出願日　平成18年4月28日 (2006.4.28)

(71)出願人　000004329
日本ビクター株式会社
神奈川県横浜市神奈川区守屋町３丁目１２番地
(74)代理人　100083806
弁理士　三好　秀和
(74)代理人　100101247
弁理士　高橋　俊一
(72)発明者　穂積　芳子
神奈川県横浜市神奈川区守屋町３丁目１２番地　日本ビクター株式会社内

Ｆターム(参考)　5C059 MA04 MA05 MA23 MC11 ME01 ME06 NN01 SS07 SS14 TC03
5J064 BA09 BA16 BB01 BC16

(54)【発明の名称】画像符号化装置および画像符号化プログラム

(57)【要約】
【課題】画像符号化処理において、係数の可変長符号化における演算量を削減し、符号化処理を高速化する。
【解決手段】コード変数算出部５２は、ジグザグスキャン部５１で生成された係数列における各係数がゼロであるか非ゼロであるかに基づいて、ゼロ係数と非ゼロ係数の配置を特定するためのコード変数を算出する。符号化部５４の第１符号化部５４ａ、第２符号化部５４ｂ、・・・、第ｎ符号化部５４ｎは、コード変数に応じて、それぞれ個別の符号化処理を行う。
【選択図】図２



【特許請求の範囲】
【請求項１】
画像をブロック単位に分割して符号化処理を行う画像符号化装置において、
予測画像と原画像との残差成分を直交変換量子化して生成された係数を高周波成分からジグザグスキャン順に並べ替えて係数列を生成する係数列生成手段と、
この係数列生成手段で生成された前記係数列における各係数がゼロ係数であるか非ゼロ係数であるかに基づいて、前記ゼロ係数と前記非ゼロ係数の配置を特定するためのコード変数を算出するコード変数算出手段と、
前記係数列の非ゼロ係数に続くゼロ係数の連続数を示すゼロラン数と、前記係数列の最初の非ゼロ係数以降のゼロ係数の数を示すゼロ数を、前記コード変数に対応して予め設定されている符号化コードとそのコード長とを用いて可変長符号化する符号化手段と
を備えることを特徴とする画像符号化装置。
【請求項２】
画像をブロック単位に分割して符号化処理を行う画像符号化装置において、
予測画像と原画像との残差成分を直交変換量子化して生成された係数を高周波成分からジグザグスキャン順に並べ替えて係数列を生成する係数列生成手段と、
この係数列生成手段で生成された前記係数列を複数の分割係数列に分割し、これら分割係数列ごとに、各係数がゼロ係数であるか非ゼロ係数であるかに基づいて、前記ゼロ係数と前記非ゼロ係数の配置を特定するためのコード変数を算出するコード変数算出手段と、
前記分割係数列ごとに、前記分割係数列の非ゼロ係数に続くゼロ係数の連続数を示すゼロラン数と、前記分割係数列の最初の非ゼロ係数以降のゼロ係数の数を示すゼロ数を、前記コード変数に対応して予め設定されている符号化コードとそのコード長とを用いて可変長符号化して符号化データを生成する符号化手段と、
この符号化手段手段により生成された、すべての前記分割係数列についての前記符号化データを統合する統合手段と
を備えることを特徴とする画像符号化装置。
【請求項３】
画像をブロック単位に分割して符号化処理を行う画像符号化プログラムにおいて、
予測画像と原画像との残差成分を直交変換量子化して生成された係数を高周波成分からジグザグスキャン順に並べ替えて係数列を生成するステップと、
この係数列生成手段で生成された前記係数列における各係数がゼロ係数であるか非ゼロ係数であるかに基づいて、前記ゼロ係数と前記非ゼロ係数の配置を特定するためのコード変数を算出するステップと、
前記係数列の非ゼロ係数に続くゼロ係数の連続数を示すゼロラン数と、前記係数列の最初の非ゼロ係数以降のゼロ係数の数を示すゼロ数を、前記コード変数に対応して予め設定されている符号化コードとそのコード長とを用いて可変長符号化するステップと
をコンピュータに実行させるための画像符号化プログラム。
【請求項４】
画像をブロック単位に分割して符号化処理を行う画像符号化プログラムにおいて、
予測画像と原画像との残差成分を直交変換量子化して生成された係数を高周波成分からジグザグスキャン順に並べ替えて係数列を生成するステップと、
この係数列生成手段で生成された前記係数列を複数の分割係数列に分割し、これら分割係数列ごとに、各係数がゼロ係数であるか非ゼロ係数であるかに基づいて、前記ゼロ係数と前記非ゼロ係数の配置を特定するためのコード変数を算出するステップと、
前記分割係数列ごとに、前記分割係数列の非ゼロ係数に続くゼロ係数の連続数を示すゼロラン数と、前記分割係数列の最初の非ゼロ係数以降のゼロ係数の数を示すゼロ数を、前記コード変数に対応して予め設定されている符号化コードとそのコード長とを用いて可変長符号化して符号化データを生成するステップと、
この符号化手段手段により生成された、すべての前記分割係数列についての前記符号化データを統合するステップと

をコンピュータに実行させるための画像符号化プログラム。

【発明の詳細な説明】

【技術分野】

【０００１】

本発明は、動画像を効率的に伝送するために、より少ない符号量で符号化する高能率の画像符号化装置および画像符号化プログラムに関する。

【背景技術】

【０００２】

従来の画像符号化装置にあっては、予測画像と原画像との残差成分を直交変換量子化して生成した係数を、規格で決められたシンタックスに基づいて可変長符号化している。

【０００３】

仮に、輝度成分のブロックの大きさを１６×１６画素、色差成分のブロックの大きさを８×８画素、係数の直交変換を４×４の単位で行うとすると、係数の数は全部で３８４個となる。これを４×４、つまり１６個ずつに区切って可変長符号化する。

【０００４】

例えば、Ｈ．２６４／ＭＰＥＧ－４　ＡＶＣ（以降、Ｈ．２６４／ＡＶＣと呼ぶ）の符号化においては、この１６個の係数をまず高周波成分からジグザグスキャン順に並べ替えてから、最初の非ゼロ係数を探し、非ゼロ係数の値やゼロ係数の並び数、非ゼロ係数の絶対値が１かどうかなどを順次判定し、可変長符号化する必要がある。

【０００５】

そのため、係数の可変長符号化処理では、非ゼロ係数とゼロ係数とがどのように分布しているかを調べるための条件判定文が多くなり、ＤＳＰ等へ実装した場合に多くの演算量を必要とする。

【０００６】

また、非ゼロ係数が多くなれば、演算量が増える。一般には、インター符号化と比較してイントラ符号化した場合の方が、非ゼロ係数が多く分布する。また、ビットレートを上げると非ゼロ係数の数が増加する。さらに、Ｈ．２６４／ＡＶＣのシンタックスで規定されている符号へ変換するためのテーブル参照が多く、これも実装時の負荷となっている。

【０００７】

従来の画像符号化装置としては、例えば特許文献１に記載されたものが報告されている。この画像符号化装置では、ジグザグスキャンして並べ替えた係数列について、非ゼロ係数とゼロ係数の分布により、非ゼロ係数の数を求め、ゼロラン数とレベル値とを組み合わせてメモリ上に格納し、この組み合わせた値からＶＬＣテーブルを参照して符号化を行っている。

【０００８】

ここで、従来の画像符号化装置における可変長符号化処理について説明する。ここでは、４×４画素の単位で直交変換された１６個の係数を、Ｈ．２６４／ＡＶＣの規格に従って可変長符号化する場合について説明する。

【０００９】

図１７は従来の画像符号化装置における可変長符号化処理を説明するためのフローチャートである。まず、図１８（ａ）に示す４×４ブロックの係数を、図１８（ｂ）に示すジグザグスキャン順に並べ替える（ステップＳ１０）。図１９（ａ）はジグザグスキャン前の係数列を示す図、図１９（ｂ）はジグザグスキャン後の係数列を示す図である。

【００１０】

そして、各係数$C_{00}'$～$C_{15}'$について判定を行う（ステップＳ２０）。１つの係数のループについて説明すると、まず、その係数が非ゼロ係数かゼロ係数かを判定し（ステップＳ３０）、ゼロ係数でない（ステップＳ３０：ＮＯ）場合は、その非ゼロ係数の係数値（レベル値）を記録する（ステップＳ６０）。そして、非ゼロ係数の数を更新し（ステップＳ７０）、ループを終了する（ステップＳ８０）。

【００１１】

また、符号化には非ゼロ係数に続くゼロ係数の連続数（ゼロラン数）が必要となる。このため、係数がゼロ係数である（ステップＳ３０：ＹＥＳ）場合は、その係数の前までに非ゼロ係数があるかどうかを判定し（ステップＳ４０）、非ゼロ係数がある（ステップＳ４０：ＹＥＳ）場合、ゼロラン数を更新して記録する（ステップＳ５０）。非ゼロ係数がない（ステップＳ４０：ＮＯ）場合、ループを終了する（ステップＳ８０）。

【００１２】

１６個の係数についてこのループが終わった後、非ゼロ係数の最初に連続する絶対値１の係数の数（先行１の数）を符号化して、それらの係数の符号とともに出力する（ステップＳ９０）。

【００１３】

また、ステップＳ６０で記録したレベル値、ステップＳ５０で記録したゼロラン数、最初の非ゼロ係数以降のゼロ係数の数（ゼロ数）を規格に従って符号化して出力する（ステップＳ１００～Ｓ１２０）。

【００１４】

図１９（ｃ）は係数列の一例を示す図、図１９（ｄ）は図１９（ｃ）に示す係数列でのゼロラン数とレベル値、およびゼロラン数とレベル値の組み合わせ数を示す表図である。

【００１５】

図１９（ｃ）に示す係数列では、非ゼロ係数が３つとなるので、ステップＳ５０，Ｓ６０において、３つのレベル値とゼロラン数を記録する。そして、Ｓ１１０においてゼロラン数を符号化して出力する際には、ゼロラン数により規定のテーブルを参照し、テーブルに規定の符号化コードを、規定のコード長で出力する。

【００１６】

また、ステップＳ９０，Ｓ１２０においても同様に、それぞれ先行１の数、ゼロ数により規定のテーブルを参照し、テーブルに規定の符号化コードを、規定のコード長で出力する。

【特許文献１】特開２００１－３０８７１５号公報

【発明の開示】

【発明が解決しようとする課題】

【００１７】

上述した可変長符号化処理では、ステップＳ３０，Ｓ４０の処理において、条件判定文が多用されるために、ＤＳＰ等に実装する場合に演算量が多くなるという問題があった。

【００１８】

また、係数列からゼロラン数等のデータを取得して別のメモリ上にストアした後に、ステップＳ９０～Ｓ１２０の処理の際に、これらのデータをロードするために、実装時にメモリへのアクセス動作が多く発生する。

【００１９】

さらに、ステップＳ９０，Ｓ１１０，Ｓ１２０の処理において、参照するテーブルをその都度切り替えるために、テーブルへのアクセス動作が多く発生し、処理に時間がかかるという問題があった。

【００２０】

本発明は上記に鑑みてなされたもので、係数の可変長符号化における演算量を削減し、符号化処理を高速化することができる画像符号化装置および画像符号化プログラムを提供することを目的とする。

【課題を解決するための手段】

【００２１】

本発明の画像符号化装置は、画像をブロック単位に分割して符号化処理を行う画像符号化装置において、予測画像と原画像との残差成分を直交変換量子化して生成された係数を高周波成分からジグザグスキャン順に並べ替えて係数列を生成する係数列生成手段と、この係数列生成手段で生成された前記係数列における各係数がゼロ係数であるか非ゼロ係数であるかに基づいて、前記ゼロ係数と前記非ゼロ係数の配置を特定するためのコード変数

を算出するコード変数算出手段と、前記係数列の非ゼロ係数に続くゼロ係数の連続数を示すゼロラン数と、前記係数列の最初の非ゼロ係数以降のゼロ係数の数を示すゼロ数を、前記コード変数に対応して予め設定されている符号化コードとそのコード長とを用いて可変長符号化する符号化手段とを備えることを特徴とする。

【００２２】

また、本発明の画像符号化装置は、画像をブロック単位に分割して符号化処理を行う画像符号化装置において、予測画像と原画像との残差成分を直交変換量子化して生成された係数を高周波成分からジグザグスキャン順に並べ替えて係数列を生成する係数列生成手段と、この係数列生成手段で生成された前記係数列を複数の分割係数列に分割し、これら分割係数列ごとに、各係数がゼロ係数であるか非ゼロ係数であるかに基づいて、前記ゼロ係数と前記非ゼロ係数の配置を特定するためのコード変数を算出するコード変数算出手段と、前記分割係数列ごとに、前記分割係数列の非ゼロ係数に続くゼロ係数の連続数を示すゼロラン数と、前記分割係数列の最初の非ゼロ係数以降のゼロ係数の数を示すゼロ数を、前記コード変数に対応して予め設定されている符号化コードとそのコード長とを用いて可変長符号化して符号化データを生成する符号化手段と、この符号化手段手段により生成された、すべての前記分割係数列についての前記符号化データを統合する統合手段とを備えることを特徴とする。

【００２３】

また、本発明の画像符号化プログラムは、画像をブロック単位に分割して符号化処理を行う画像符号化プログラムにおいて、予測画像と原画像との残差成分を直交変換量子化して生成された係数を高周波成分からジグザグスキャン順に並べ替えて係数列を生成するステップと、この係数列生成手段で生成された前記係数列における各係数がゼロ係数であるか非ゼロ係数であるかに基づいて、前記ゼロ係数と前記非ゼロ係数の配置を特定するためのコード変数を算出するステップと、前記係数列の非ゼロ係数に続くゼロ係数の連続数を示すゼロラン数と、前記係数列の最初の非ゼロ係数以降のゼロ係数の数を示すゼロ数を、前記コード変数に対応して予め設定されている符号化コードとそのコード長とを用いて可変長符号化するステップとをコンピュータに実行させることを特徴とする。

【００２４】

また、本発明の画像符号化プログラムは、画像をブロック単位に分割して符号化処理を行う画像符号化プログラムにおいて、予測画像と原画像との残差成分を直交変換量子化して生成された係数を高周波成分からジグザグスキャン順に並べ替えて係数列を生成するステップと、この係数列生成手段で生成された前記係数列を複数の分割係数列に分割し、これら分割係数列ごとに、各係数がゼロ係数であるか非ゼロ係数であるかに基づいて、前記ゼロ係数と前記非ゼロ係数の配置を特定するためのコード変数を算出するステップと、前記分割係数列ごとに、前記分割係数列の非ゼロ係数に続くゼロ係数の連続数を示すゼロラン数と、前記分割係数列の最初の非ゼロ係数以降のゼロ係数の数を示すゼロ数を、前記コード変数に対応して予め設定されている符号化コードとそのコード長とを用いて可変長符号化して符号化データを生成するステップと、この符号化手段手段により生成された、すべての前記分割係数列についての前記符号化データを統合するステップとをコンピュータに実行させることを特徴とする。

【発明の効果】

【００２５】

本発明によれば、符号化する係数のゼロ係数と非ゼロ係数の配置を特定するためのコード変数を算出し、このコード変数に応じた個別の可変長符号化処理を行うので、係数の非ゼロ／ゼロ判定における条件判定文にかかる演算量や、符号化の際のテーブル参照にかかるメモリアクセスを削減することができ、係数の可変長符号化における演算量を削減し、符号化処理を高速化することができる。

【発明を実施するための最良の形態】

【００２６】

以下、本発明を実施するための最良の形態について、図面を参照して説明する。

【００２７】
（第１の実施の形態）
図１は本発明の第１の実施の形態に係る画像符号化装置の構成を示すブロック図である。図１に示すように、第１の実施の形態に係る画像符号化装置は、フレームメモリ１、演算器２、直交変換部３、量子化部４、可変長符号化部５、バッファ６、逆量子化部７、逆直交変換部８、演算器９、フレームメモリ１０、動き検出部１１、動き補償部１２、画面内符号化部１３、スイッチ１４を備える。
【００２８】
フレームメモリ１は、入力画像信号を蓄積する。演算器２は、フレーム内符号化（以下、イントラ符号化という）が行われる場合、入力画像信号から画面内符号化部１３により生成される予測信号を減算し、残差信号を求める。フレーム間符号化（以下、インター符号化という）が行われる場合、演算器２は、入力画像信号から動き補償部１２により生成される予測信号を減算し、残差信号を求める。
【００２９】
直交変換部３は、演算器２で生成された残差信号に離散コサイン変換等の直交変換を施す。量子化部４は、直交変換が施された変換係数の量子化処理を行う。可変長符号化部５は、量子化部４で生成された量子化データに可変長符号化を施し、符号化ビットストリームを出力する。
【００３０】
逆量子化部７は、量子化部４で生成された量子化データの逆量子化を行う。逆直交変換部８は、逆量子化のなされたデータに逆直交変換を施し、復号された残差信号を出力する。
【００３１】
演算器９は、イントラ符号化が行われる場合、逆直交変換部８からの復号された残差信号と、画面内符号化部１３より供給される予測信号とを加算し、復号画像信号を生成する。また、インター符号化が行われる場合、演算器９は、逆直交変換部８からの復号された残差信号と、動き補償部１２より供給される予測信号とを加算し、復号画像信号を生成する。フレームメモリ１０は、演算器９からの復号画像信号を蓄積する。
【００３２】
動き検出部１１は、フレームメモリ１０を参照して動きベクトルを検出し、検出された動きベクトルを動き補償部１２に供給する。動き補償部１２、動きベクトルを用いて動き補償処理を行う。画面内符号化部１３は、入力画像に対してフレーム内予測処理を行う。
【００３３】
スイッチ１４はポート１４ａ，１４ｂ，１４ｃを備え、ポート１４ｂ－１４ａが接続されたときには演算器２と画面内符号化部１３とが接続され、ポート１４ｃ－１４ａが接続されたときには演算器２と動き補償部１２とが接続される。
【００３４】
図２は図１に示す画像符号化装置の可変長符号化部の構成を示すブロック図である。図２に示すように可変長符号化部５は、ジグザグスキャン部５１、コード変数算出部５２、スイッチ５３、符号化部５４、ビット列生成部５６、ＣＡＶＬＣテーブル５７、制御部５８を備える。また、符号化部５４は、第１符号化部５４ａ、第２符号化部５４ｂ、・・・、第ｎ符号化部５４ｎを備える。
【００３５】
ジグザグスキャン部５１は、量子化部４から出力される量子化された変換係数をジグザグスキャン順に並べ替える。
【００３６】
コード変数算出部５２は、ジグザグスキャン部５１で生成された係数列における各係数がゼロであるか非ゼロであるかに基づいて、ゼロ係数と非ゼロ係数の配置を特定するためのコード変数を算出する。スイッチ５３は、コード変数算出部５２で算出されたコード変数に応じて、係数列の出力先を切り替える。

【００３７】
符号化部５４の第１符号化部５４ａ、第２符号化部５４ｂ、・・・、第ｎ符号化部５４ｎは、コード変数に対応して設けられ、それぞれ個別の可変長符号化処理を行う。ビット列生成部５６は、符号化部５４から出力される符号化コードとコード長とをビット列（符号化ビットストリーム）として生成し、生成したビット列をバッファ６に出力する。
【００３８】
ＣＡＶＬＣテーブル５７は、ＣＡＶＬＣ（Ｃｏｎｔｅｘｔ－Ａｄａｐｔｉｖｅ Ｖａｒｉａｂｌｅ Ｌｅｎｇｔｈ Ｃｏｄｉｎｇ）で用いられるテーブルであり、先行１の数を符号化するためのテーブル、ゼロ数を符号化するためのテーブル、ゼロラン数を符号化するためのテーブルを含み、各テーブルには、符号化コードと、符号化コードの長さ（コード長）を示すデータとが保存されている。制御部５８は、可変長符号化部５に設けられた各部を制御する。
【００３９】
次に、第１の実施の形態に係る画像符号化装置の動作を説明する。
【００４０】
入力された画像信号はフレームメモリ１に蓄えられ、フレーム内またはフレーム間の信号が符号化される。
【００４１】
イントラ符号化ではフレーム内の画像信号のみが独立して符号化され、インター符号化では、直前、直後のフレームを参照フレームとする予測信号が生成され、予測誤差が符号化される。
【００４２】
イントラ符号化においては、入力画像信号はフレームメモリ１に蓄えられた後、演算器２と画面内符号化部１３とに入力され、画面内符号化部１３により生成される予測信号はスイッチ１４のポート１４ｂ－１４ａを介して演算器２に供給される。演算器２では、入力画像信号と画面内符号化部１３により生成される予測信号との残差信号が求められ、得られた残差信号は直交変換部３に供給される。直交変換部３では、離散コサイン変換等の直交変換が施され、得られた変換係数が量子化部４に供給される。
【００４３】
量子化部４では、変換係数の量子化がなされ、生成された量子化データは、可変長符号化部５に供給される。そして、可変長符号化部５で量子化データに可変長符号化が施された後、符号化ビットストリームが出力され、バッファ６へ保存または外部へ伝送される。
【００４４】
また、量子化部４で生成された量子化データは、逆量子化部７に供給されて逆量子化がなされ、その逆量子化のなされたデータは逆直交変換部８に供給されて逆直交変換が施され、復号された残差信号となる。
【００４５】
逆直交変換部８で得られた復号された残差信号は、演算器９に供給され、画面内符号化部１３より供給される予測信号が加算されて復号画像信号となり、フレームメモリ１０に蓄積される。
【００４６】
一方、インター符号化においては、まず、入力画像信号が動き検出部１１に入力される。動き検出部１１では、フレームメモリ１０を参照して動きベクトルを検出し、検出された動きベクトルは動き補償部１２に供給される。動き補償部１２では、動きベクトルを用いて動き補償処理が行われ、予測信号が生成される。
【００４７】
この予測信号は、スイッチ１４のポート１４ｃ－１４ａを介して演算器２に供給される。演算器２では、入力画像信号と動き補償部１２により生成される予測信号との残差信号が求められ、得られた残差信号は直交変換部３に供給される。直交変換部３では、離散コサイン変換等の直交変換が施され、得られた変換係数が量子化部４に供給される。

【００４８】
量子化部４では、変換係数の量子化がなされ、生成された量子化データは、可変長符号化部５に供給される。そして、可変長符号化部５で量子化データに可変長符号化が施された後、符号化ビットストリームが出力され、バッファ６へ保存または外部へ伝送される。
【００４９】
また、量子化部４で生成された量子化データは、逆量子化部７に供給されて逆量子化がなされ、その逆量子化のなされたデータは逆直交変換部８に供給されて逆直交変換が施され、復号された残差信号となる。
【００５０】
逆直交変換部８で得られた復号された残差信号は、演算器９に供給され、動き補償部１２より供給される予測信号が加算されて復号画像信号となり、フレームメモリ１０に蓄積される。
【００５１】
次に、可変長符号化部５における係数の可変長符号化処理について説明する。図４は図２に示す可変長符号化部５における可変長符号化処理を説明するためのフローチャートである。
【００５２】
まずステップＳ２１０において、制御部５８は、係数の非ゼロ位置情報を示す１６ビットのコード変数を用意し、これを０に初期化するようにコード変数算出部５２を制御する。図５はコード変数を説明するための図である。図５（ａ）に示すジグザグスキャン後の係数列に対して、図５（ｂ）に示すように、各ビットの０／１を係数のゼロ／非ゼロに対応させる。
【００５３】
次に、ステップＳ２２０では、制御部５８は、量子化部４から入力される係数を、高周波成分からジグザグスキャン順に並べ替えるようにジグザグスキャン部５１を制御する。
【００５４】
そして、制御部５８は、各係数$C_{00}'$～$C_{15}'$について判定を行うようにコード変数算出部５２を制御する（ステップＳ２３０）。１つの係数のループについて説明すると、まず、ステップＳ２４０において、コード変数算出部５２は、その係数が非ゼロ係数かゼロ係数かを判定し、ゼロ係数でない（ステップＳ２４０：ＮＯ）場合は、ステップＳ２５０において、コードのビットをＯＮにする。ゼロ係数である（ステップＳ２４０：ＹＥＳ）場合はコードのビットをＯＮせず、ループを終了する（ステップＳ２６０）。
【００５５】
これにより、１６個の係数の非ゼロ位置情報が１６ビットのコード変数で表現できる。図５（ｃ）に示すように、係数$C_{15}'$のみが非ゼロ係数である場合はコード＝１、係数$C_{14}'$のみが非ゼロ係数である場合はコード＝２となる。
【００５６】
次に、Ｓ２７０では、制御部５８は、コード変数算出部５２で算出したコード変数がいくつであるかを判断し、ジグザグスキャン部５１からコード変数算出部５２を介して出力される係数列の出力先を切り替えるようにスイッチ５３を制御する。
【００５７】
そして、制御部５８は、例えば、コード＝０の場合は第１符号化部５４ａ、コード＝１の場合は第２符号化部５４ｂに係数列を出力し、個別の可変長符号化処理を行わせる（ステップＳ２８０ａ，Ｓ２８０ｂ，・・・）。
【００５８】
ここで、コード＝１の場合の第２符号化部５４ｂにおける可変長符号化処理について説明する。コード＝１となるのは、図６に示すように、係数$C_{15}'$のみが非ゼロ係数である場合である。図７は第２符号化部５４ｂにおける可変長符号化処理を説明するためのフローチャートである。
【００５９】

まず、ステップＳ３１０において、第２符号化部５４ｂは、係数$C_{15}'$の係数値（レベル値）の絶対値が１であるかどうかを判定する。Ｈ．２６４／ＡＶＣでは、非ゼロ係数の最初に連続する絶対値１の係数の数（先行１の数）を符号化することによって符号化効率を上げているので、この判定が必要となる。

【００６０】

係数$C_{15}'$のレベル値の絶対値が１である（ステップＳ３１０：ＹＥＳ）場合、ステップＳ３２０において、第２符号化部５４ｂは、ＣＡＶＬＣテーブル５７を参照して、先行１の数を符号化するためのテーブルに規定されている先行１の数＝１に対応する符号化コードを、テーブルに規定のコード長で出力する。また、第２符号化部５４ｂは、係数$C_{15}'$の正負の符号についても符号化して符号化コードを出力する。

【００６１】

係数$C_{15}'$のレベル値の絶対値が１でない（ステップＳ３１０：ＮＯ）場合、ステップＳ３３０において、第２符号化部５４ｂは、ＣＡＶＬＣテーブル５７を参照して、先行１の数を符号化するためのテーブルに規定されている先行１の数＝０に対応する符号化コードを、テーブルに規定のコード長で出力する。その後、ステップＳ３４０では、第２符号化部５４ｂは、係数$C_{15}'$のレベル値を符号化して符号化コードを出力する。

【００６２】

次に、ステップＳ３５０において、第２符号化部５４ｂは、ゼロラン数を符号化して符号化コードを出力するが、コード＝１の場合は、ゼロラン数の符号化コードは出力不要である。

【００６３】

そして、ステップＳ３６０において、第２符号化部５４ｂは、ゼロ数（＝０）に対応して予め設定されている符号化コード（＝１）を、予め設定されているコード長（＝１ビット）で出力する。

【００６４】

その後、ビット列生成部５６は、第２符号化部５４ｂから出力された符号化コードとコード長とをビット列（符号化ビットストリーム）として生成し、生成したビット列をバッファ６に出力する。

【００６５】

ここで、従来の可変長符号化処理においては、ゼロラン数、ゼロ数を符号化して出力する際、ＣＡＶＬＣテーブルを参照していた。図８はＣＡＶＬＣにおいてゼロ数を符号化するために用いられるテーブルを示す図である。

【００６６】

従来の可変長符号化処理では、例えばゼロ数が０、非ゼロ係数の個数が１のとき、図８に示すテーブルを参照して、符号化コード＝１、コード長＝１ビットを取得していた。第１の実施の形態の画像符号化装置では、第２符号化部５４ｂに、ゼロ数（＝０）に対応して符号化コード（＝１）、コード長（＝１ビット）が予め設定されているため、テーブルを参照する必要がない。ゼロラン数についても同様に、符号化コード、コード長が予め設定され、テーブルを参照することなく符号化することができる。

【００６７】

次に、コード＝２の場合の第３符号化部５４ｃにおける可変長符号化処理について説明する。コード＝２となるのは、図９に示すように、係数$C_{14}'$のみが非ゼロ係数である場合である。図１０は第３符号化部５４ｃにおける可変長符号化処理を説明するためのフローチャートである。

【００６８】

まず、ステップＳ４１０において、第３符号化部５４ｃは、係数$C_{14}'$のレベル値の絶対値が１であるかどうかを判定する。

【００６９】

係数$C_{14}'$のレベル値の絶対値が１である（ステップＳ４１０：ＹＥＳ）場合、ステップＳ４２０において、第３符号化部５４ｃは、ＣＡＶＬＣテーブル５７を参照して、先

行１の数を符号化するためのテーブルに規定されている先行１の数＝１に対応する符号化コードを、テーブルに規定のコード長で出力する。また、第３符号化部５４ｃは、係数$C_{14}$´の正負の符号についても符号化して符号化コードを出力する。

【００７０】

係数$C_{14}$´のレベル値の絶対値が１でない（ステップＳ４１０：ＮＯ）場合、ステップＳ４３０において、第３符号化部５４ｃは、ＣＡＶＬＣテーブル５７を参照して、先行１の数を符号化するためのテーブルに規定されている先行１の数＝０に対応する符号化コードを、テーブルに規定のコード長で出力する。その後、ステップＳ４４０では、第３符号化部５４ｃは、係数$C_{14}$´のレベル値を符号化して符号化コードを出力する。

【００７１】

次に、ステップＳ４５０において、第３符号化部５４ｃは、ゼロラン数（＝１）に対応して予め設定されている符号化コードを、予め設定されているコード長で出力する。そして、ステップＳ４６０において、第３符号化部５４ｃは、ゼロ数（＝１）に対応して予め設定されている符号化コードを、予め設定されているコード長で出力する。

【００７２】

次に、コード＝１１の場合の第１２符号化部５４１における可変長符号化処理について説明する。コード＝１１となるのは、図１１に示すように、係数$C_{12}$´，$C_{14}$´，$C_{15}$´が非ゼロ係数である場合である。図１２は第１２符号化部５４１における可変長符号化処理を説明するためのフローチャートである。

【００７３】

まず、ステップＳ５１０において、第１２符号化部５４１は、係数$C_{12}$´のレベル値の絶対値が１であるかどうかを判定する。

【００７４】

係数$C_{12}$´のレベル値の絶対値が１である（ステップＳ５１０：ＹＥＳ）場合、ステップＳ５２０に進む。係数$C_{12}$´のレベル値の絶対値が１でない（ステップＳ５１０：ＮＯ）場合、ステップＳ５９０において、第１２符号化部５４１は、ＣＡＶＬＣテーブル５７を参照して、先行１の数を符号化するためのテーブルに規定されている先行１の数＝０に対応する符号化コードを、テーブルに規定のコード長で出力する。その後、ステップＳ６００では、第１２符号化部５４１は、係数$C_{12}$´，$C_{14}$´，$C_{15}$´のレベル値をそれぞれ符号化して符号化コードを出力する。

【００７５】

次に、ステップＳ５２０において、第１２符号化部５４１は、係数$C_{14}$´のレベル値の絶対値が１であるかどうかを判定する。

【００７６】

係数$C_{14}$´のレベル値の絶対値が１である（ステップＳ５２０：ＹＥＳ）場合、ステップＳ５３０に進む。係数$C_{14}$´のレベル値の絶対値が１でない（ステップＳ５２０：ＮＯ）場合、ステップＳ５７０において、第１２符号化部５４１は、ＣＡＶＬＣテーブル５７を参照して、先行１の数を符号化するためのテーブルに規定されている先行１の数＝１に対応する符号化コードを、テーブルに規定のコード長で出力する。その後、ステップＳ５８０では、第１２符号化部５４１は、係数$C_{14}$´，$C_{15}$´のレベル値をそれぞれ符号化して符号化コードを出力する。

【００７７】

次に、ステップＳ５３０において、第１２符号化部５４１は、係数$C_{15}$´のレベル値の絶対値が１であるかどうかを判定する。

【００７８】

係数$C_{15}$´のレベル値の絶対値が１である（ステップＳ５３０：ＹＥＳ）場合、ステップＳ５４０に進む。係数$C_{14}$´のレベル値の絶対値が１でない（ステップＳ５３０：ＮＯ）場合、ステップＳ５５０において、第１２符号化部５４１は、ＣＡＶＬＣテーブル５７を参照して、先行１の数を符号化するためのテーブルに規定されている先行１の数＝２に対応する符号化コードを、テーブルに規定のコード長で出力する。その後、ステップ

Ｓ５６０では、第１２符号化部５４１は、係数$C_{14}'$のレベル値を符号化して出力する。

【００７９】

次いで、ステップＳ５４０において、第１２符号化部５４１は、ＣＡＶＬＣテーブル５７を参照して、先行１の数を符号化するためのテーブルに規定されている先行１の数＝３に対応する符号化コードを、テーブルに規定のコード長で出力する。

【００８０】

次に、ステップＳ６１０において、第１２符号化部５４１は、ゼロラン数に対応して予め設定されている符号化コードを、予め設定されているコード長で出力する。そして、ステップＳ６２０において、第１２符号化部５４１は、ゼロ数に対応して予め設定されている符号化コードを、予め設定されているコード長で出力する。

【００８１】

このように、第１の実施の形態によれば、係数の可変長符号化を行う際に、係数と非ゼロ係数の配置を特定するためのコード変数を算出し、このコード変数に対応する個別の符号化処理を行うので、コード変数により非ゼロ係数の位置やゼロラン数が確定し、非ゼロ係数の位置情報やゼロ係数の連続数を取得するための演算量を削減することができる。また、ゼロラン数、ゼロ数を符号化して出力する際にＣＡＶＬＣテーブルを参照する必要がなく、テーブル参照の回数を減らすことができ、符号化処理を高速化することができる。

【００８２】

ところで、係数が１６個ある場合にはコード変数は１６ビットの値となり、０～６５３５の範囲の値を持つ。しかし、このすべてのパターンについて個別の符号化処理を行うようにするのは困難である。そこで、直交変換の性質から係数が低周波成分に集中することを利用して、頻度の高いパターンのみについて個別の符号化処理を行うことが可能である。例えば、$C_{12}'$～$C_{15}'$のみに非ゼロ係数が分布する場合のみ個別の符号化処理を行うようにすることができる。

【００８３】

図３は第１の実施の形態に係る画像符号化装置の変形例の可変長符号化部の構成を示すブロック図である。図３に示すように可変長符号化部５ａは、図２に示す可変長符号化部５に対し、符号化部５４を符号化部５５に置き換えた構成である。符号化部５５は、第１符号化部５４ａ、第２符号化部５４ｂ、・・・、第１６符号化部５４ｐ、範囲外符号化部５５ａを備える。

【００８４】

可変長符号化部５ａでは、例えば、$C_{12}'$～$C_{15}'$のみに非ゼロ係数が分布する場合、つまりコード変数が０～１５である場合、第１符号化部５４ａ、第２符号化部５４ｂ、・・・、第１６符号化部５４ｐにおいて、上記に説明したような個別の可変長符号化処理を行う。コード変数が１５より大きい値である場合は、範囲外符号化部５５ａにおいて、図１６のフローチャートに示した従来の符号化処理の手順で可変長符号化処理が行われる。

【００８５】

非ゼロ係数が低周波に集中する場合、例えば低レート符号化やインター符号化では、この変形例の画像符号化装置により、符号化処理を高速に行うことができる。

【００８６】

（第２の実施の形態）

次に、第２の実施の形態について説明する。第２の実施の形態に係る画像符号化装置の構成は、図１に示す第１の実施の形態に係る画像符号化装置の構成と同様である。

【００８７】

図１３は第２の実施の形態に係る画像符号化装置の可変長符号化部の構成を示すブロック図である。図１３に示すように可変長符号化部５ｂは、ジグザグスキャン部６１、コード変数算出部６２、スイッチ６３、符号化部６４、ビット列生成部６５、ビット列格納部６６、ビット列統合部６７、ＣＡＶＬＣテーブル６８、制御部６９を備える。また、符号

化部６４は、第１符号化部６４ａ、第２符号化部６４ｂ、・・・、第１６符号化部６４ｐを備える。

【００８８】

ジグザグスキャン部６１は、量子化部４から出力される量子化された変換係数をジグザグスキャン順に並べ替える。

【００８９】

コード変数算出部６２は、ジグザグスキャン部６１で生成された係数列を複数の分割係数列に分割し、分割係数列ごとに、各係数がゼロであるか非ゼロであるかに基づいて、ゼロ係数と非ゼロ係数の配置を特定するためのコード変数を算出する。スイッチ６３は、コード変数算出部６２で算出されたコード変数に応じて、係数列の出力先を切り替える。

【００９０】

符号化部６４の第１符号化部６４ａ、第２符号化部６４ｂ、・・・、第１６符号化部６４ｐは、コード変数に対応して設けられ、それぞれ個別の可変長符号化処理を行う。ビット列生成部６５は、符号化部６４から出力される符号化コードとコード長とをビット列として生成し、生成したビット列をビット列格納部６６に出力する。

【００９１】

ビット列格納部６６は、ビット列生成部６５生成された分割係数列ごとのビット列を格納する。ビット列統合部６７は、ビット列格納部６６に格納された、すべての分割係数列についてのビット列を統合し、統合したビット列（符号化ビットストリーム）をバッファ６へ出力する。

【００９２】

ＣＡＶＬＣテーブル６８は、ＣＡＶＬＣで用いられるテーブルであり、先行１の数を符号化するためのテーブル、ゼロ数を符号化するためのテーブル、ゼロラン数を符号化するためのテーブルを含み、各テーブルには、符号化コードと、符号化コードの長さ（コード長）を示すデータとが保存されている。制御部６９は、可変長符号化部５ｂに設けられた各部を制御する。

【００９３】

次に、可変長符号化部５ｂにおける係数の可変長符号化処理について説明する。以下、１６個の係数を含む係数列を、４個の係数からなる４つの分割係数列に分割する例について説明するが、分割数はこれに限らない。

【００９４】

図１５は図１３に示す可変長符号化部５ｂにおける可変長符号化処理を説明するためのフローチャートである。まずステップＳ７１０において、制御部６９は、係数の非ゼロ位置情報を示す１６ビットのコード変数を用意し、これを０に初期化するようにコード変数算出部６２を制御する。

【００９５】

次に、ステップＳ７２０では、制御部６９は、量子化部４から入力される係数を、高周波成分からジグザグスキャン順に並べ替えるようにジグザグスキャン部６１を制御する。

【００９６】

そして、制御部６９は、各係数$C_{00}'$～$C_{15}'$を、図１４に示すように、上位から４個の係数からなる４つの分割係数列に分割し、各分割係数列の係数について判定を行うようにコード変数算出部６２を制御する（ステップＳ７３０，Ｓ７４０）。ループ２は分割係数列についてのループであり、ループ１は各分割係数列の係数についてのループである。

【００９７】

ループ１における１つの係数のループについて説明すると、まず、ステップＳ７５０において、コード変数算出部６２は、その係数が非ゼロ係数かゼロ係数かを判定し、ゼロ係数でない（ステップＳ７５０：ＮＯ）場合は、ステップＳ７６０において、コードのビットをＯＮにする。ゼロ係数である（ステップＳ７５０：ＹＥＳ）場合はコードのビットをＯＮせず、ループ１を終了する（ステップＳ７７０）。

【００９８】
これにより、１つの分割係数列について、４個の係数の非ゼロ位置情報が４ビットのコード変数で表現できる。図１４に示すように、例えば、最初の分割係数列について、係数$C_{01}'$のみが非ゼロ係数である場合はコード＝４となる。
【００９９】
次に、Ｓ７８０では、制御部６９は、コード変数算出部６２で算出したコード変数がいくつであるかを判断し、ジグザグスキャン部６１からコード変数算出部６２を介して出力される分割係数列の出力先を切り替えるようにスイッチ６３を制御する。
【０１００】
そして、制御部６９は、例えば、コード＝０の場合は第１符号化部５４ａ、コード＝１の場合は第２符号化部５４ｂに係数列を出力し、個別の可変長符号化処理を行わせる（ステップＳ７９０ａ，Ｓ７９０ｂ，・・・）。ここで、第１符号化部５４ａ、第２符号化部５４ｂ、・・・における符号化処理は、第１の実施の形態で説明した手順と同様に行われる。
【０１０１】
その後、ビット列生成部６５は、符号化部６４から出力された符号化コードとコード長とをビット列（符号化ビットストリーム）として生成し、生成したビット列をビット列格納部６６に出力する。ここで、ゼロラン数は分割係数列間にまたがるデータであるため、ステップＳ８００において、各分割係数列の符号化処理が終わる毎にデータの更新を行い、ループ２を終了する（ステップＳ８１０）。
【０１０２】
ループ２が分割数分（４回）終了すると、ステップＳ８２０において、制御部６９は、ビット列格納部６６に格納された、すべての分割係数列についてのビット列を符号化処理の順序に従って統合し、統合したビット列（符号化ビットストリーム）をバッファ６へ出力するようにビット列格納部６６を制御する。
【０１０３】
高レート符号化やイントラ符号化では非ゼロ係数の分布が広がるので、第１の実施の形態に係る画像符号化装置では十分な高速化の効果が得られない場合がある。第２の実施の形態では、係数列を上位より任意の数に分割して可変長符号化処理を行うので、非ゼロ係数が多く含まれる係数列においても演算量を削減することができ、高速な符号化処理が可能となる。
【０１０４】
なお、本発明は上記装置の機能をコンピュータに実現させるためのプログラムを含むものである。このプログラムは、記録媒体から読み取られてコンピュータに取り込まれてもよいし、通信ネットワークを介して伝送されてコンピュータに取り込まれてもよい。
【０１０５】
図１６は本発明に係る画像符号化プログラムを実行可能なパーソナルコンピュータ（ＰＣ）の構成を示す図である。
【０１０６】
図１６において、ＣＰＵ１０１は、ＲＯＭ１０２に記憶されているプログラム、または記憶部１０８からＲＡＭ１０３にロードされたプログラムに従って、画像符号化処理や各種の処理を実行する。ＲＡＭ１０３にはまた、ＣＰＵ１０１が画像符号化処理や各種の処理を実行する上において必要なデータなども適宜記憶される。
【０１０７】
ＣＰＵ１０１、ＲＯＭ１０２、およびＲＡＭ１０３は、バス１０４を介して相互に接続されている。また、このバス１０４には、入出力インタフェース１０５も接続されている。
【０１０８】
入出力インタフェース１０５には、キーボード、マウスなどからなる入力部１０６、ディスプレイやスピーカなどからなる出力部１０７、ハードディスクなどから構成される

記憶部１０８、モデム、ターミナルアダプタなどから構成される通信部１０９が接続されている。通信部１０９は、インターネットを含む通信ネットワークを介しての通信処理を行う。

【０１０９】

入出力インタフェース１０５にはまた、必要に応じてドライブ１１０が接続され、磁気ディスク１１１、光ディスク１１２、光磁気ディスク１１３、もしくは、半導体メモリ１１４などが適宜装着され、それらから読み出された、画像符号化処理や各種の処理を実行するためのプログラムが、必要に応じて記憶部１０８にインストールされる。

【産業上の利用可能性】

【０１１０】

本発明は、係数部の可変長符号化を行う画像符号化装置として、カムコーダやビデオレコーダ、監視カメラやＴＶ会議システム等に適用できる。また、ＰＣ上のエンコーダプログラムに適用できる。

【図面の簡単な説明】

【０１１１】

【図１】本発明の第１の実施の形態に係る画像符号化装置の構成を示すブロック図である。

【図２】図１に示す画像符号化装置の可変長符号化部の構成を示すブロック図である。

【図３】第１の実施の形態に係る画像符号化装置の変形例の可変長符号化部の構成を示すブロック図である。

【図４】図２に示す可変長符号化部５における可変長符号化処理を説明するためのフローチャートである。

【図５】コード変数を説明するための図である。

【図６】コード＝１を示す図である。

【図７】第２符号化部５４ｂにおける可変長符号化処理を説明するためのフローチャートである。

【図８】ＣＡＶＬＣにおいてゼロ数を符号化するために用いられるテーブルを示す図である。

【図９】コード＝２を示す図である。

【図１０】第３符号化部５４ｃにおける可変長符号化処理を説明するためのフローチャートである。

【図１１】コード＝１１を示す図である。

【図１２】第１２符号化部５４ｌにおける可変長符号化処理を説明するためのフローチャートである。

【図１３】第２の実施の形態に係る画像符号化装置の可変長符号化部の構成を示すブロック図である。

【図１４】係数列の分割を説明するための図である。

【図１５】図１３に示す可変長符号化部５ｂにおける可変長符号化処理を説明するためのフローチャートである。

【図１６】本発明に係る画像符号化プログラムを実行可能なパーソナルコンピュータ（ＰＣ）の構成を示す図である。

【図１７】従来の画像符号化装置における可変長符号化処理を説明するためのフローチャートである。

【図１８】ジグザグスキャンを説明するための図である。

【図１９】（ａ）はジグザグスキャン前の係数列を示す図、（ｂ）はジグザグスキャン後の係数列を示す図、（ｃ）は係数列の一例を示す図、（ｄ）は（ｃ）に示す係数列でのゼロラン数とレベル値、およびゼロラン数とレベル値の組み合わせ数を示す表図である。

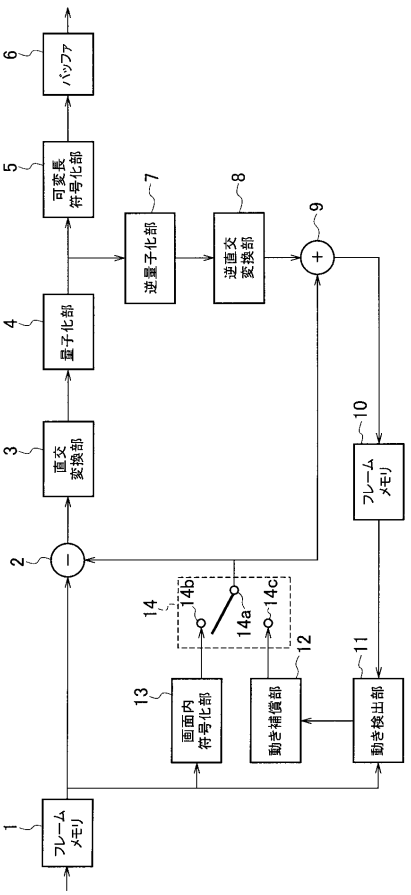
【符号の説明】

【０１１２】

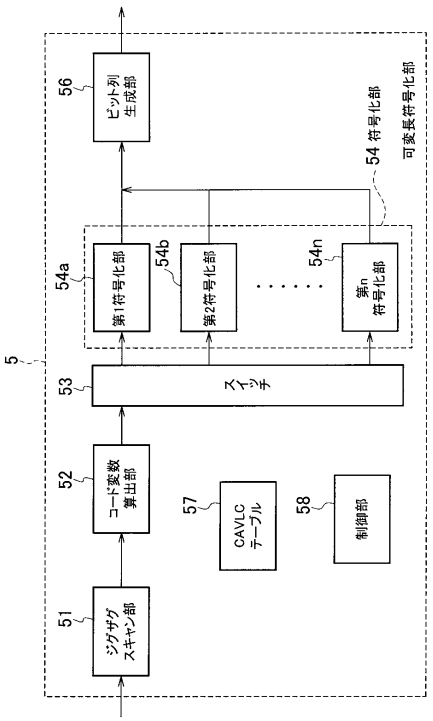
１，１０　フレームメモリ

２，９　　演算器
３　　直交変換部
４　　量子化部
５，５ａ，５ｂ　　可変長符号化部
６　　バッファ
７　　逆量子化部
８　　逆直交変換部
１１　　動き検出部
１２　　動き補償部
１３　　画面内符号化部
１４　　スイッチ
５１，６１　　ジグザグスキャン部
５２，６２　　コード変数算出部
５３，６３　　スイッチ
５４，５５，６４　　符号化部
５４ａ，６４ａ　　第１符号化部
５４ｂ，６４ｂ　　第２符号化部
５４ｐ，６４ｐ　　第１６符号化部
５５ａ　　範囲外符号化部
５６，６５　　ビット列生成部
６６　　ビット列格納部
６７　　ビット列統合部
５７，６８　　ＣＡＶＬＣテーブル
５８，６９　　制御部

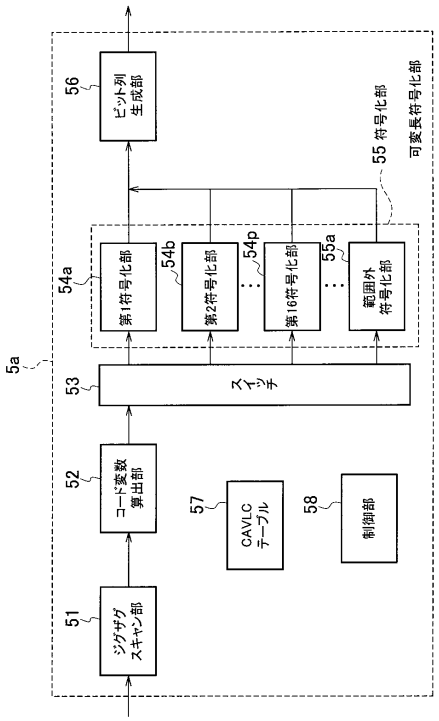
【図１】



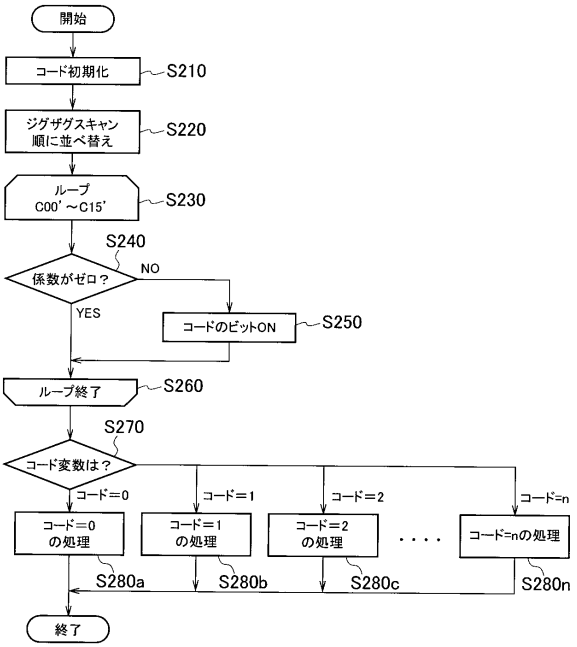

【図２】



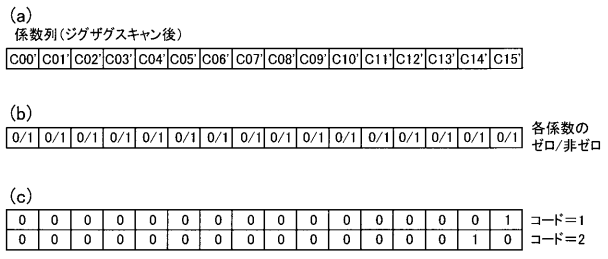
【図３】


ビット列生成部
56
55 符号化部
可変長符号化部
第1符号化部
54a
第2符号化部
54b
第16符号化部
54p
範囲外符号化部
55a
5a
スイッチ
53
コード変数算出部
52
CAVLCテーブル
57
制御部
58
ジグザグスキャン部
51

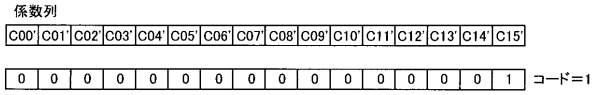
【図４】


開始
コード初期化
S210
ジグザグスキャン順に並べ替え
S220
ループ C00'～C15'
S230
S240
係数がゼロ？
NO
YES
コードのビットON
S250
ループ終了
S260
S270
コード変数は？
コード＝0
コード＝1
コード＝2
コード=n
コード＝0の処理
コード＝1の処理
コード＝2の処理
コード=nの処理
S280a
S280b
S280c
S280n
終了

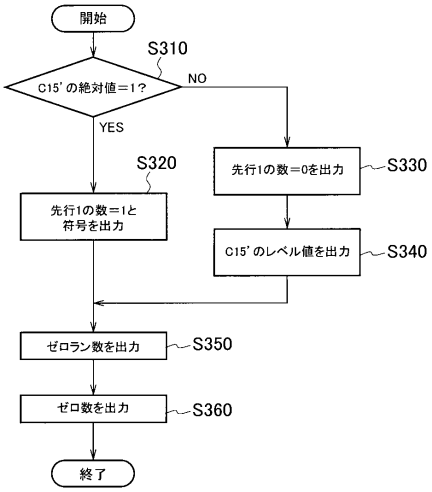
【図５】


(a)
係数列（ジグザグスキャン後）
C00' C01' C02' C03' C04' C05' C06' C07' C08' C09' C10' C11' C12' C13' C14' C15'
(b)
0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1
各係数のゼロ/非ゼロ
(c)
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
コード＝1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
コード＝2

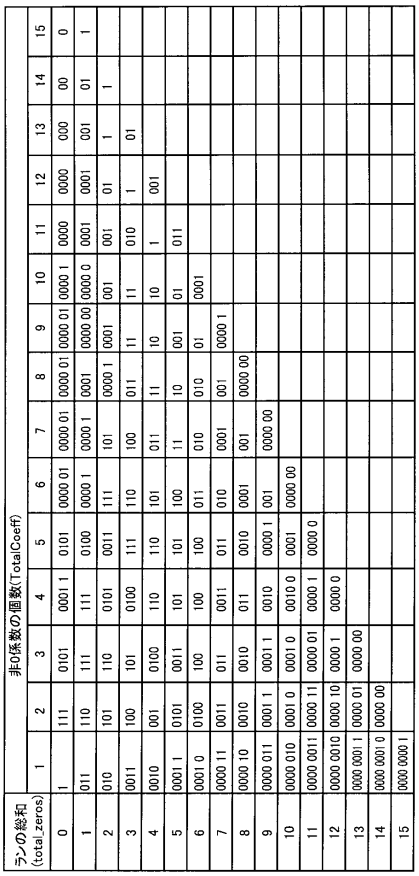
【図６】


係数列
C00' C01' C02' C03' C04' C05' C06' C07' C08' C09' C10' C11' C12' C13' C14' C15'
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
コード＝1

【図７】

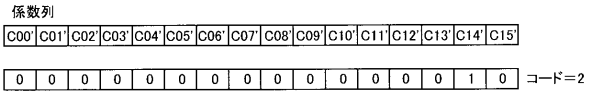

開始
S310
C15' の絶対値＝1？
NO
YES
S320
先行1の数＝1と符号を出力
先行1の数＝0を出力
S330
C15' のレベル値を出力
S340
ゼロラン数を出力
S350
ゼロ数を出力
S360
終了

【図８】

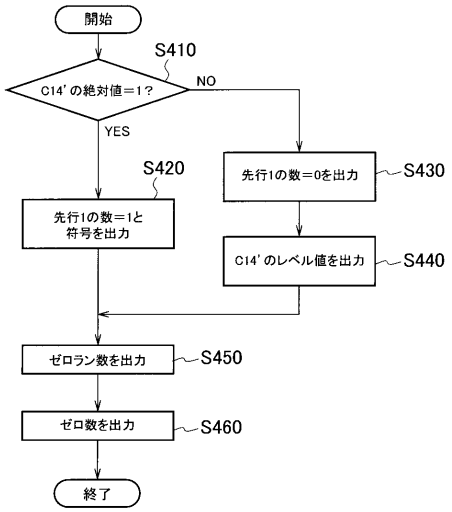


| ランの総和 (total_zeros) | 非0係数の個数(TotalCoeff) 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 111 | 0101 | 0001 1 | 0101 | 0000 01 | 0000 01 | 0000 01 | 0000 01 | 0000 1 | 0000 | 0000 | 000 | 00 | 0 |
| 1 | 011 | 110 | 111 | 111 | 0100 | 0000 1 | 0000 1 | 0001 | 0000 00 | 0000 0 | 0001 | 0001 | 001 | 01 | 1 |
| 2 | 010 | 101 | 110 | 0101 | 0011 | 111 | 101 | 0000 1 | 0001 | 001 | 001 | 01 | 1 | 1 | |
| 3 | 0011 | 100 | 101 | 0100 | 111 | 110 | 100 | 011 | 11 | 11 | 010 | 1 | 01 | | |
| 4 | 0010 | 001 | 0100 | 110 | 110 | 101 | 011 | 11 | 10 | 10 | 1 | 001 | | | |
| 5 | 0001 1 | 0101 | 0011 | 101 | 101 | 100 | 11 | 10 | 001 | 01 | 011 | | | | |
| 6 | 0001 0 | 0100 | 100 | 100 | 100 | 011 | 010 | 010 | 01 | 0001 | | | | | |
| 7 | 0000 11 | 0011 | 011 | 0011 | 011 | 010 | 0001 | 001 | 0000 1 | | | | | | |
| 8 | 0000 10 | 0010 | 0010 | 011 | 0010 | 0001 | 001 | 0000 00 | | | | | | | |
| 9 | 0000 011 | 0001 1 | 0001 1 | 0010 | 0000 1 | 001 | 0000 00 | | | | | | | | |
| 10 | 0000 010 | 0001 0 | 0001 0 | 0010 0 | 0001 | 0000 00 | | | | | | | | | |
| 11 | 0000 0011 | 0000 11 | 0000 01 | 0000 1 | 0000 0 | | | | | | | | | | |
| 12 | 0000 0010 | 0000 10 | 0000 1 | 0000 0 | | | | | | | | | | | |
| 13 | 0000 0001 1 | 0000 01 | 0000 00 | | | | | | | | | | | | |
| 14 | 0000 0001 0 | 0000 00 | | | | | | | | | | | | | |
| 15 | 0000 0000 1 | | | | | | | | | | | | | | |

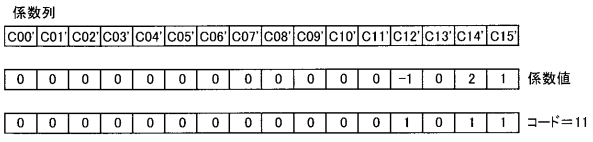
【図９】



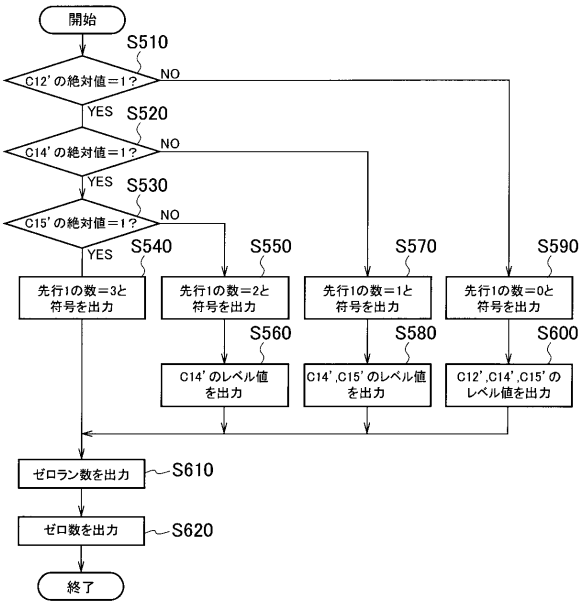

【図１０】



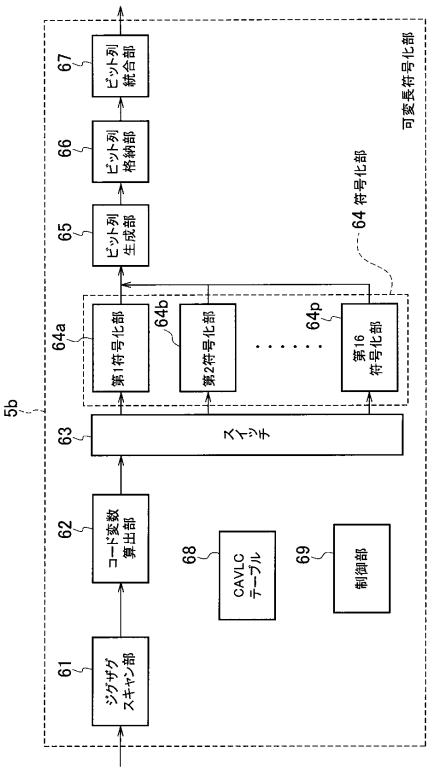

【図１１】



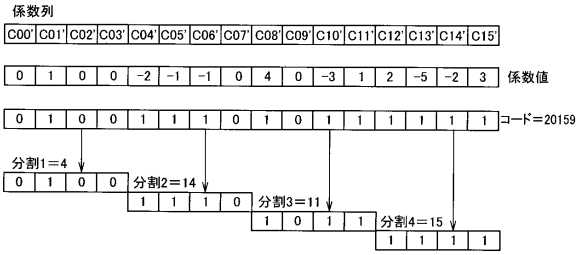

【図１２】



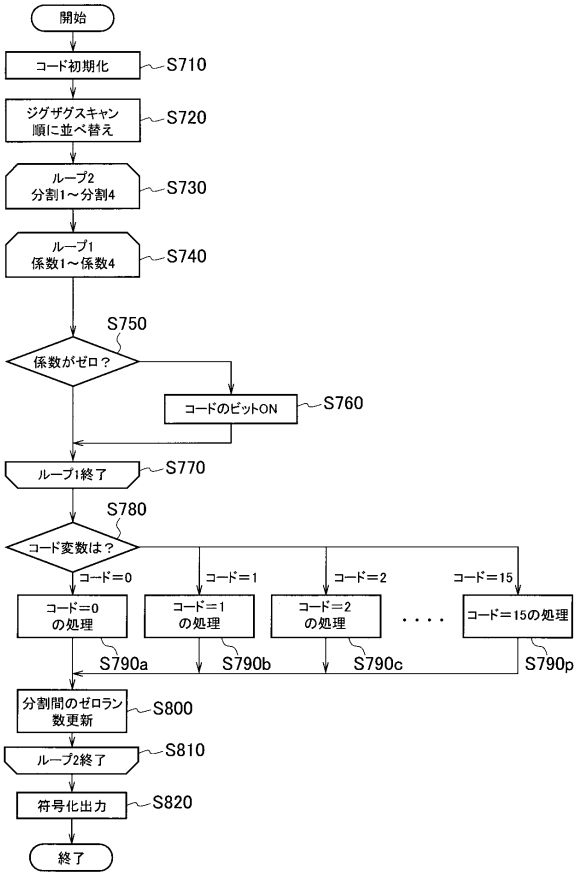

【図１３】



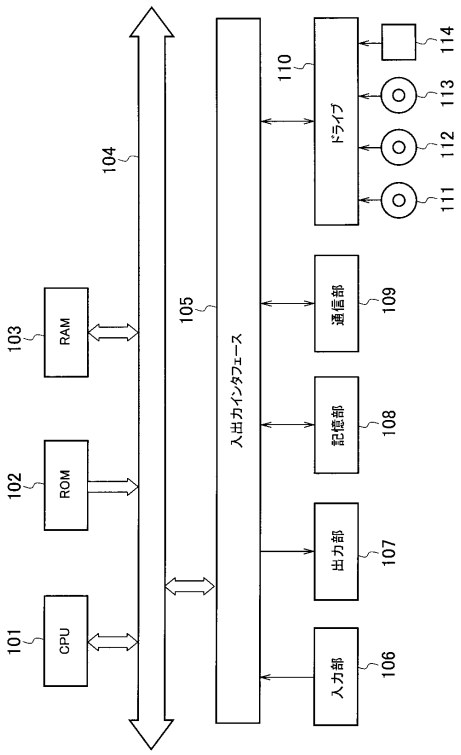
【図１４】



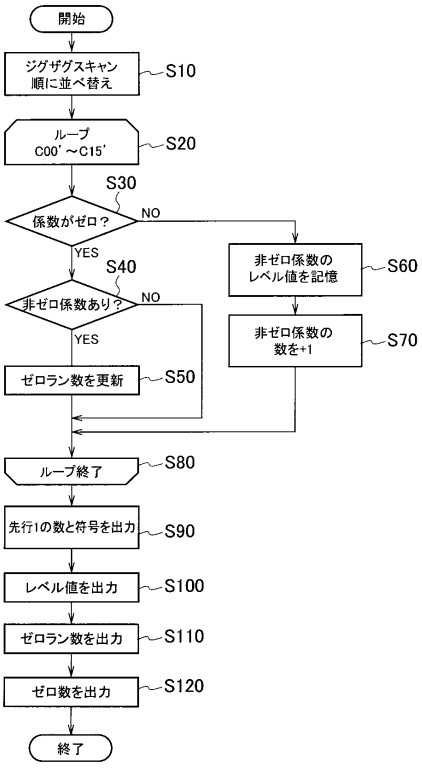

【図１５】




【図１６】




【図１７】



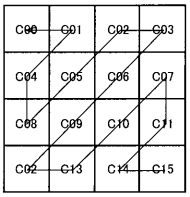
【図１８】

(a)

4×4ブロックの係数

| | | | |
|---|---|---|---|
| C00 | C01 | C02 | C03 |
| C04 | C05 | C06 | C07 |
| C08 | C09 | C10 | C11 |
| C02 | C13 | C14 | C15 |

(b)




ジグザグスキャン

【図１９】

(a) 係数列（ジグザグスキャン前）

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| C00 | C01 | C02 | C03 | C04 | C05 | C06 | C07 | C08 | C09 | C10 | C11 | C12 | C13 | C14 | C15 |

(b) 係数列（ジグザグスキャン後）

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| C00' | C01' | C02' | C03' | C04' | C05' | C06' | C07' | C08' | C09' | C10' | C11' | C12' | C13' | C14' | C15' |

(c) 係数列の例

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | -1 | 0 | 2 | 1 |

(d)

| ゼロラン数 | レベル値 | （ゼロラン+レベル）の数 |
|---|---|---|
| 1 | -1 | 3 |
| 0 | 2 | |
| 0 | 1 | |