

(19) 日本国特許庁 (JP)

(12) 特 許 公 報 (B2)

(11) 特許番号

特許第4793320号
(P4793320)

(45) 発行日 平成23年10月12日 (2011.10.12)

(24) 登録日 平成23年8月5日 (2011.8.5)

(51) Int.Cl.		F I	
HO 4 N	7/30	(2006.01)	HO 4 N 7/133 Z
HO 4 N	1/41	(2006.01)	HO 4 N 1/41 B

請求項の数 8 (全 51 頁)

(21) 出願番号	特願2007-131866 (P2007-131866)	(73) 特許権者	000002185
(22) 出願日	平成19年5月17日 (2007.5.17)		ソニー株式会社
(65) 公開番号	特開2008-288883 (P2008-288883A)		東京都港区港南1丁目7番1号
(43) 公開日	平成20年11月27日 (2008.11.27)	(74) 代理人	100082131
審査請求日	平成22年3月17日 (2010.3.17)		弁理士 稲本 義雄
		(72) 発明者	安藤 勝俊
			東京都港区港南1丁目7番1号 ソニー株
			式会社内
		(72) 発明者	福原 隆浩
			東京都港区港南1丁目7番1号 ソニー株
			式会社内
		審査官	畑中 高行

最終頁に続く

(54) 【発明の名称】 情報処理装置および方法

(57) 【特許請求の範囲】

【請求項 1】

画像データの周波数成分をリフティング演算により帯域毎に分割する分析フィルタ処理手段と、

前記分析フィルタ処理手段により、帯域毎に分割された前記周波数成分を可変長符号化する符号化手段と、

前記分析フィルタ処理手段を制御し、画像データが2ライン入力される毎に、前記リフティング演算の、実行可能な演算を実行させ、さらに、前記符号化手段を制御し、前記演算が終了してから、次の画像データ2ラインが入力されるまでの間に、符号化可能な周波数成分を可変長符号化させる制御手段と

を備える情報処理装置。

【請求項 2】

前記制御手段は、

前記分析フィルタ処理手段に、実行可能な演算のうち、より高域の周波数成分を生成する演算を、より先に実行させ、

前記符号化手段に、符号化可能な周波数成分のうち、より低域の周波数成分であり、かつ、同じ帯域の中では、より先に生成された周波数成分を、より先に可変長符号化させる

請求項 1 に記載の情報処理装置。

【請求項 3】

10

20

前記制御手段は、前記符号化手段に、常に、前記リフティング演算により直前に生成された最低域の周波数成分より先に生成された周波数成分を可変長符号化させる

請求項 2 に記載の情報処理装置。

【請求項 4】

前記符号化手段により前記周波数成分が符号化されて得られた符号化データを保持する符号化データ保持手段をさらに備え、

前記制御手段は、前記リフティング演算により最低域の周波数成分が生成されるよりも先に、前記最低域の周波数成分に対応する、前記最低域の周波数成分よりも高域の周波数成分を符号化させ、得られた符号化データを前記符号化データ保持手段に保持させ、前記最低域の周波数成分が生成され符号化され出力された後に、前記符号化データ保持手段に保持されている前記符号化データを所定の順に出力させる

10

請求項 2 に記載の情報処理装置。

【請求項 5】

前記分析フィルタ処理手段により帯域毎に分割された周波数成分を保持する周波数成分保持手段をさらに備え、

前記符号化手段は、前記周波数成分保持手段に保持されている前記周波数成分を可変長符号化する

請求項 1 に記載の画像処理装置。

【請求項 6】

情報処理装置の情報処理方法であって、

20

画像データの周波数成分をリフティング演算により帯域毎に分割する分析フィルタ処理部を制御し、前記画像データが 2 ライン入力される毎に、前記リフティング演算の、実行可能な演算を実行させ、

帯域毎に分割された前記周波数成分を可変長符号化する符号化部を制御し、前記演算が終了してから、次の画像データ 2 ラインが入力されるまでの間に、符号化可能な周波数成分を可変長符号化させる

ステップを含む情報処理方法。

【請求項 7】

画像データの周波数成分をリフティング演算により帯域毎に分割する分析フィルタ処理手段と、

30

前記分析フィルタ処理手段により、帯域毎に分割された前記周波数成分を可変長符号化する符号化手段と、

前記符号化手段を制御し、入力された複数ラインの画像データを用いて行われた、前記リフティング演算の実行可能な演算が終了してから、次の画像データの複数ラインが入力されるまでの間に、符号化可能な周波数成分を可変長符号化させる制御手段と

を備える情報処理装置。

【請求項 8】

情報処理装置の情報処理方法であって、

リフティング演算によって帯域毎に分割された画像データの周波数成分を可変長符号化する符号化部を制御し、分析フィルタ処理部において前記画像データが 2 ライン入力される毎に行われる、前記リフティング演算の、実行可能な演算が終了してから、次の画像データ 2 ラインが入力されるまでの間に、符号化可能な周波数成分を可変長符号化させる

40

ステップを含む情報処理方法。

【発明の詳細な説明】

【技術分野】

【0001】

本発明は、情報処理装置および方法に関し、特に、ウェーブレット変換処理および符号化処理を効率よく行うようにし、負荷をより低減させることができるようにした情報処理装置および方法に関する。

【背景技術】

50

【 0 0 0 2 】

従来の代表的な画像圧縮方式として、ISO (International Standards Organization) によって標準化されたJPEG(Joint Photographic Experts Group)がある。これは離散コサイン変換(Discrete Cosine Transform ; DCT)を用い、比較的高いビットが割り当てられる場合には、良好な符号化画像および復号画像を供することが知られている。

【 0 0 0 3 】

近年では画像をフィルタバンクと呼ばれるハイパス・フィルタとローパス・フィルタとを組み合わせたフィルタによって複数の帯域に分割し、帯域毎に符号化を行う方式の研究が盛んになっている。その中でも、ウェーブレット変換符号化は、DCT変換で問題になる高圧縮でのブロック歪みが無いことから、DCTに代わる新たな技術として有力視されている。

10

【 0 0 0 4 】

2 0 0 1 年 1 月に国際標準化が完了したJPEG 2 0 0 0 は、このウェーブレット変換に高性能なエントロピ符号化(ビットプレーン単位のビット・モデリングと算術符号化)を組み合わせた方式を採用しており、JPEGに比べて符号化効率の大きな改善を実現している。

【 0 0 0 5 】

ウェーブレット変換処理(例えば特許文献 1 参照)は基本的に画像データを入力して、水平方向のフィルタリングと垂直方向のフィルタリングを行いながら、低域成分を階層的に分割する手段を用いる。

【 0 0 0 6 】

このウェーブレット変換処理により画像データが変換された係数データ(周波数成分)を、元の画像データに変換するウェーブレット逆変換処理は、最上位の分割レベルから最下位の分割レベルまで高域成分と低域成分を合成フィルタ処理しながら、最終的に画像を復元する処理を行う。

20

【 0 0 0 7 】

このようなウェーブレット変換・ウェーブレット逆変換を利用した符号化システムは、例えばTV会議システムやビデオゲームシステム等のように、画像データの伝送を行うシステムに利用されることが多い。つまり、送信側において、画像データがウェーブレット変換され、得られた係数データがエントロピ符号化されて符号化データとして受信側に伝送される。受信側においては、取得した符号化データがエントロピ復号されて、得られた係数データがウェーブレット逆変換されて元の画像データが復元される。このような処理の流れが一般的である。

30

【 0 0 0 8 】

このような画像伝送システムにおいては、画像データの伝送を低遅延で行うことが望ましい場合が多く、ウェーブレット変換処理、エントロピ符号化、ウェーブレット逆変換処理、エントロピ復号処理等の各処理に対して遅延時間の低減が求められる。

【 0 0 0 9 】

ところで、ウェーブレット変換処理およびエントロピ符号化処理を行うエンコーダをソフトウェアプログラムにより実現するソフトウェアエンコーダや、エントロピ復号処理やウェーブレット逆変換処理を行うデコーダをソフトウェアプログラムにより実現するソフトウェアデコーダがある。

40

【 0 0 1 0 】

これらのソフトウェアエンコーダやソフトウェアデコーダは、そのソフトウェアプログラムがパーソナルコンピュータ等の情報処理装置において実行されることにより、エンコーダやデコーダとしての機能が実現される。

【 0 0 1 1 】

【特許文献 1】特開平 1 0 - 2 8 3 3 4 2 号公報

【発明の開示】

【発明が解決しようとする課題】

【 0 0 1 2 】

50

しかしながら、通常の場合、エンコーダには多様な処理が含まれており、その処理の数は、ソフトウェアプログラムを実行するCPU (Central Processing Unit) の数よりも多い。つまり、ソフトウェアエンコーダの場合、1つのCPUにおいて多様な処理を実行することになる。換言すれば1つの演算処理部を複数の処理で共有することになる。

【0013】

従って、エンコーダ機能を実現する各処理を効率よく行うように適切に実行させなければ、CPUの使用効率が低下したり、処理に必要なメモリ容量が増大したりして、負荷が増大してしまう恐れがあった。負荷の増大は、コストや遅延時間の増大に繋がる恐れもある。

【0014】

本発明は、このような従来の実情に鑑みて提案されたものであり、ウェーブレット変換処理および符号化処理をより効率よく実行させることにより、ハードウェアに負荷をより低減させることができるようにするものである。

【課題を解決するための手段】

【0015】

本発明の第1の側面は、画像データの周波数成分をリフティング演算により帯域毎に分割する分析フィルタ処理手段と、前記分析フィルタ処理手段により、帯域毎に分割された前記周波数成分を可変長符号化する符号化手段と、前記分析フィルタ処理手段を制御し、画像データが2ライン入力される毎に、前記リフティング演算の、実行可能な演算を実行させ、さらに、前記符号化手段を制御し、前記演算が終了してから、次の画像データ2ラインが入力されるまでの間に、符号化可能な周波数成分を可変長符号化させる制御手段とを備える情報処理装置である。

【0016】

前記制御手段は、前記分析フィルタ処理手段に、実行可能な演算のうち、より高域の周波数成分を生成する演算を、より先に実行させ、前記符号化手段に、符号化可能な周波数成分のうち、より低域の周波数成分であり、かつ、同じ帯域の中では、より先に生成された周波数成分を、より先に可変長符号化させることができる。

【0017】

前記制御手段は、前記符号化手段に、常に、前記リフティング演算により直前に生成された最低域の周波数成分より先に生成された周波数成分を可変長符号化させることができる。

【0018】

前記符号化手段により前記周波数成分が符号化されて得られた符号化データを保持する符号化データ保持手段をさらに備え、前記制御手段は、前記リフティング演算により最低域の周波数成分が生成されるよりも先に、前記最低域の周波数成分に対応する、前記最低域の周波数成分よりも高域の周波数成分を符号化させ、得られた符号化データを前記符号化データ保持手段に保持させ、前記最低域の周波数成分が生成され符号化され出力された後に、前記符号化データ保持手段に保持されている前記符号化データを所定の順に出力させることができる。

前記分析フィルタ処理手段により帯域毎に分割された周波数成分を保持する周波数成分保持手段をさらに備え、前記符号化手段は、前記周波数成分保持手段に保持されている前記周波数成分を可変長符号化することができる。

【0019】

本発明の第1の側面はまた、情報処理装置の情報処理方法であって、画像データの周波数成分をリフティング演算により帯域毎に分割する分析フィルタ処理部を制御し、前記画像データが2ライン入力される毎に、前記リフティング演算の、実行可能な演算を実行させ、帯域毎に分割された前記周波数成分を可変長符号化する符号化部を制御し、前記演算が終了してから、次の画像データ2ラインが入力されるまでの間に、符号化可能な周波数成分を可変長符号化させるステップを含む情報処理方法である。

【0021】

10

20

30

40

50

本発明の第2の側面は、画像データの周波数成分をリフティング演算により帯域毎に分割する分析フィルタ処理手段と、前記分析フィルタ処理手段により、帯域毎に分割された前記周波数成分を可変長符号化する符号化手段と、前記符号化手段を制御し、入力された複数ラインの画像データを用いて行われた、前記リフティング演算の実行可能な演算が終了してから、次の画像データの複数ラインが入力されるまでの間に、符号化可能な周波数成分を可変長符号化させる制御手段とを備える情報処理装置である。

本発明の第2の側面はまた、情報処理装置の情報処理方法であって、リフティング演算によって帯域毎に分割された画像データの周波数成分を可変長符号化する符号化部を制御し、分析フィルタ処理部において前記画像データが2ライン入力される毎に行われる、前記リフティング演算の、実行可能な演算が終了してから、次の画像データ2ラインが入力されるまでの間に、符号化可能な周波数成分を可変長符号化させるステップを含む情報処理方法である。

10

【0022】

本発明の第1の側面においては、画像データの周波数成分をリフティング演算により帯域毎に分割する分析フィルタ処理部が制御され、画像データが2ライン入力される毎に、リフティング演算の、実行可能な演算が実行され、帯域毎に分割された周波数成分を可変長符号化する符号化部が制御され、演算が終了してから、次の画像データ2ラインが入力されるまでの間に、符号化可能な周波数成分が可変長符号化される。

【0023】

本発明の第2の側面においては、リフティング演算によって帯域毎に分割された画像データの周波数成分を可変長符号化する符号化部が制御され、分析フィルタ処理部において画像データが2ライン入力される毎に行われる、リフティング演算の、実行可能な演算が終了してから、次の画像データ2ラインが入力されるまでの間に、符号化可能な周波数成分が可変長符号化される。

20

【発明の効果】

【0024】

本発明によれば、ウェーブレット変換処理および符号化処理を行うことができる。特に、より高効率にウェーブレット変換処理および符号化処理を行うことができ、ハードウェアにかかる負荷をより低減させることができる。

【発明を実施するための最良の形態】

30

【0033】

以下、本発明の実施の形態について説明する。まず、符号化側の構成および動作について説明する。

【0034】

以下、本発明の実施の形態について説明する。まず、本発明に用いるウェーブレット変換およびウェーブレット逆変換を利用した符号化・復号システムの構成および動作について説明する。

【0035】

図1は、ソフトウェアプログラムにより構成され、画像データを符号化するソフトウェアエンコーダが有する機能を模式的に示した機能ブロック図である。図1に示される符号化部10は、ソフトウェアエンコーダであり、ソフトウェアプログラムがCPU(Central Processing Unit)により実行されることにより、ウェーブレット変換部11、途中計算用バッファ部12、係数並び替え用バッファ部13、係数並び替え部14、およびエントロピ符号化部15の機能を有する。

40

【0036】

ウェーブレット変換部11は、符号化部10に入力された画像データに対して、画像データの周波数成分をその帯域毎に分離させるウェーブレット変換処理を行う。詳細については後述するが、ウェーブレット変換部11は、分析フィルタ処理により、周波数成分を低域成分と高域成分に分離する。また、ウェーブレット変換部11は、得られた低域成分に対して分析フィルタ処理を再帰的に繰り返し行うことにより、周波数成分を帯域毎に階

50

層的に分割する。なお、以下において、この周波数成分を、ウェーブレット係数、ウェーブレット係数データ、係数データ、または係数と称する。また、以下において、周波数成分分割の階層を分割レベルと称する。

【 0 0 3 7 】

分析フィルタ処理を繰り返す際において、ウェーブレット変換部 1 1 は、次回の分析フィルタ処理に利用する係数データや画像データを途中計算用データとして途中計算用バッファ部 1 2 に保持させる。つまり、ウェーブレット変換部 1 1 は、途中計算用バッファ部 1 2 に保持させている途中計算用データ（画像データや係数データ）を読み出し、その途中計算用データを用いて、外部より入力された画像データに対してウェーブレット変換処理を行う。

10

【 0 0 3 8 】

ウェーブレット変換部 1 1 は、各分割レベルにおいて得られた高域成分（係数データ）、および、予め定められた最終的な分割レベル（最上位レベル）の低域成分（係数データ）を、係数並び替え用バッファ部 1 3 に書き込む。

【 0 0 3 9 】

係数並び替え部 1 4 は、係数並び替え用バッファ部 1 3 に書き込まれた係数データを所定の順序で読み出し、エントロピ符号化部 1 5 に供給する。詳細については後述するが、係数データは、ウェーブレット変換処理における生成順と、ウェーブレット逆変換処理における利用順とが異なる。そこで、係数並び替え部 1 4 は、係数並び替え用バッファ部 1 3 より係数データを、ウェーブレット逆変換処理において利用される順に読み出すことにより、係数データの並び替えを行う。

20

【 0 0 4 0 】

エントロピ符号化部 1 5 は、供給された係数データを、所定の方法で量子化し、例えばハフマン符号化や算術符号化といった所定のエントロピ符号化方式で符号化する。つまり、エントロピ符号化部 1 5 は、係数データを、ウェーブレット逆変換処理において利用される順に符号化する。エントロピ符号化部 1 5 は、生成した符号化データを符号化部 1 0 の外部に出力する。

【 0 0 4 1 】

図 2 は、ソフトウェアプログラムにより構成され、図 1 の符号化部 1 0 に対応するソフトウェアデコーダが有する機能を模式的に示した機能ブロック図である。この復号部 2 0 は、符号化部 1 0 において画像データが符号化されて得られた符号化データを復号し、画像データを復元する処理部であり、エントロピ復号部 2 1、係数バッファ部 2 2、およびウェーブレット逆変換部 2 3 の機能を有する。

30

【 0 0 4 2 】

符号化データが入力されると、エントロピ復号部 2 1 は、その符号化データに対して、図 1 のエントロピ符号化部 1 5 に対応する方法でエントロピ復号を行い、係数データを復元する。エントロピ復号部 2 1 は、得られた係数データを係数バッファ部 2 2 に保持させる。ウェーブレット逆変換部 2 3 は、係数バッファ部 2 2 に保持されている係数データに対して、図 1 のウェーブレット変換部 1 1 が行ったウェーブレット変換処理の逆変換処理を行い、元の画像データを復元する。

40

【 0 0 4 3 】

つまり、ウェーブレット逆変換部 2 3 は、ウェーブレット変換部 1 1 により階層的に分割された周波数成分に対して、最上位の分割レベルから順に、低域成分と高域成分を合成する合成フィルタ処理を行い、周波数成分の分割レベルを 1 つずつ下げていき、最終的にベースバンドの画像データを復元する。ウェーブレット逆変換部 2 3 は、生成した画像データを復号部 2 0 の外部に出力する。

【 0 0 4 4 】

次に、このような符号化部 1 0 および復号部 2 0 を用いるシステムにおいて利用されるウェーブレット変換処理について説明する。画像データに対するウェーブレット変換では、図 3 に概略的に示されるように、画像データを空間周波数の高い帯域と低い帯域とに分

50

割する処理を、分割の結果得られる空間周波数の低い帯域のデータに対して再帰的に繰り返す。

【 0 0 4 5 】

分析フィルタには、画像データに対して画面水平方向に分析フィルタ処理を行う水平分析フィルタと、画面垂直方向に分析フィルタ処理を行う垂直分析フィルタとがあり、各方向に対して1回ずつ分析フィルタ処理が行われることにより、画像データは4つの帯域(サブバンド)に分割される。図1のウェーブレット変換部11は、分析フィルタ処理結果の、水平方向および垂直方向のいずれにおいても空間周波数の低い帯域に対して、上述した水平方向および垂直方向の分析フィルタ処理が再帰的に繰り返す(つまり階層的に分割する)。

10

【 0 0 4 6 】

図3は、分析フィルタ処理が3回繰り返された場合の例を概略的に示す図である。図3の例では、水平方向および垂直方向の分析フィルタ処理が再帰的に3回繰り返えされることにより、1ピクチャの画像データの周波数成分(係数データ)が、10個の階層的なサブバンドに分割されている。

【 0 0 4 7 】

図3において、実線の四角および点線の角丸四角のそれぞれは、分析フィルタ処理により生成されるサブバンドを示しており、各サブバンドに表記される数字は、そのサブバンドの階層のレベル(分割レベル)を示す。つまり、ベースバンドの画像データに対して何回分析フィルタ処理することにより得られるサブバンドであるかを示している。また、各サブバンドに表記される「L」および「H」は、それぞれ低域成分および高域成分を表しており、左側が水平方向の分析フィルタ処理結果、右側が垂直方向の分析フィルタ処理結果を示している。

20

【 0 0 4 8 】

図3の例では、ベースバンドの画像データに対して1回目の分析フィルタ処理が行われて、分割レベル1の4つのサブバンド(1LL、1LH、1HL、および1HH)が生成され、そのサブバンドのうち、水平方向および垂直方向の両方に対して低域成分であるサブバンド「1LL」に対して2回目の分析フィルタ処理が行われ、分割レベル2の4つのサブバンド(2LL、2LH、2HL、および2HH)が生成され、その水平方向および垂直方向の両方に対して低域成分であるサブバンド「2LL」に対して3回目の分析フィルタ処理が行われ、分割レベル3の4つのサブバンド(3LL、3LH、3HL、および3HH)が生成されている。

30

【 0 0 4 9 】

このように、低域成分に対して繰り返し変換および分割を行うのは、図4に示されるように、より上位(低域成分)のサブバンドほど、画像のエネルギーが低域成分に集中しているためである。このように分析フィルタ処理を再帰的に処理を行い、階層的なサブバンドを生成し、空間周波数の低い帯域のデータをより小さな領域に追いついでいくことで、エントロピー符号化を行う際に効率的な圧縮符号化を可能とする。

【 0 0 5 0 】

なお、以下において、分析フィルタ処理により生成される4つのサブバンドのうち、再度分析フィルタ処理が行われる、水平方向および垂直方向の両方に対して低域成分であるサブバンド「LL」を低域サブバンドと称し、それ以上分析フィルタ処理が行われないその他のサブバンド「LH」、「HL」、および「HH」を高域サブバンドと称する。

40

【 0 0 5 1 】

このようなウェーブレット変換処理をピクチャ全体に対してまとめて行う方法もあるが、1ピクチャの画像データを数ライン毎に分割し、それぞれについてウェーブレット変換処理を互いに独立して行う方法もある。前者の場合よりも後者の場合の方が、1回のウェーブレット変換処理で処理される画像データのデータ量が少ないので、ウェーブレット変換処理の処理結果の出力開始タイミングをより早くすることができる。つまり、ウェーブレット変換処理による負荷や遅延時間を低減させることができる。

50

【 0 0 5 2 】

なお、この場合のウェーブレット変換処理の処理単位となるライン数は、ウェーブレット変換処理の、予め定められた分割レベルにおいて、最上位レベルのサブバンドの係数データ 1 ラインを得るために必要なライン数に基づく。

【 0 0 5 3 】

分析フィルタ処理により、データは 4 分割されるので、図 4 に示されるように、分割レベル 1 では 4 ラインであったものが、分割レベル 2 では 2 ラインとなり、分割レベル 3 では 1 ラインとなる。つまり、分割レベルが 1 つ上がるとともにライン数は半減する。換言すると、分割レベル 3 のウェーブレット変換処理の場合、最上位レベルのサブバンド (3 L L、3 L H、3 H L、および 3 H H) の係数データを 1 ライン得るためには、8 ラインのベースバンドの画像データが必要になる。従ってこの場合、ウェーブレット変換処理は、ベースバンドの画像データ 8 ライン以上を処理単位としてウェーブレット変換処理が行われる。

10

【 0 0 5 4 】

このように、最上位レベルの低域サブバンド「L L」の係数データを 1 ライン生成するために必要なベースバンドの画素データの集合、若しくは、その画像データより得られる係数データの集合をプレシнкт (Precinct) (またはラインブロック) と称する。

【 0 0 5 5 】

なお、図 4 に示されるように、水平方向のデータ数は、分割レベルが 1 つ上がるとともに半減する。つまり、ベースバンドの画像データの水平方向の画素数が 1 9 2 0 画素の場合、分割レベル 1 の係数データの水平方向のデータ数は 9 6 0 となり、分割レベル 2 の係数データの水平方向のデータ数は 4 8 0 となり、分割レベル 3 の係数データの水平方向のデータ数は 2 4 0 となる。

20

【 0 0 5 6 】

ウェーブレット変換処理およびウェーブレット逆変換処理は、基本的に、ピクチャ全体ではなく、このプレシнктを単位として行われる。つまり、1 ピクチャの画像データは、それぞれが複数ラインからなる複数のプレシнктに分割され、そのプレシнкт毎に複数回に分けて処理される。従って、基本的にはウェーブレット変換処理およびウェーブレット逆変換処理は、そのプレシнкт毎に完結する。例えば、1 プレシнктのベースバンドの画像データに対してウェーブレット変換処理が行われると、最低域成分 (最上位の分割レベルの係数データ) が少なくとも 1 ライン以上からなる、周波数帯域毎に階層的に分割された係数データ群が得られ、その係数データ群に対してウェーブレット逆変換処理が行われると、元の 1 プレシнкт分のベースバンドの画像データが得られる。

30

【 0 0 5 7 】

このように、ピクチャよりも細かいプレシнктを単位としてウェーブレット変換処理やウェーブレット逆変換処理を行うことにより、ウェーブレット変換処理やウェーブレット逆変換処理を行うために蓄積させる画像データや係数データのデータ量 (蓄積量) を低減させることができる。つまり、より早いタイミングで変換後の係数データや画像データを出力させることができ、ウェーブレット変換処理やウェーブレット逆変換処理による負荷や遅延時間を低減させることができる。

40

【 0 0 5 8 】

なお、さらに画像データの各ラインを複数に分割したブロックを処理単位としてウェーブレット変換処理やウェーブレット逆変換処理を行う方法も考えられるが、その場合、水平同期タイミング以上に細かい単位で制御を行わなければならない、プレシнктを処理単位とする場合よりも、制御が高速かつ複雑になり負荷が増大する恐れがある。また、ブロックを構成するライン数の最小数は、プレシнктと同様に分割レベルにより制限される。1 ピクチャの画像データは、ライン毎に、画面上のラインから順に供給されるので、ブロックを処理単位とする場合も、プレシнктを処理単位とする場合と略同程度の画像データや係数データを蓄積させる必要がある。従って、ウェーブレット変換処理やウェーブレット逆変換処理の遅延時間は、どちらの場合もほとんど変化しない。

50

【 0 0 5 9 】

なお、1 プレシントのライン数はピクチャ内において、各プレシントで互いに同一でなくてもよい。

【 0 0 6 0 】

ウェーブレット変換には例えば 5×3 フィルタや 9×7 フィルタ等の分析フィルタが用いられる。 5×3 フィルタを用いた方法は、JPEG 2 0 0 0 規格でも採用されており、少ないフィルタタップ数でウェーブレット変換を行うことができる点で、優れた方法である。

【 0 0 6 1 】

5×3 フィルタのインパルス応答 (Z 変換表現) は、下記の式 (1) および式 (2) に示す様に、低域フィルタ $H_0(z)$ と、高域フィルタ $H_1(z)$ とから構成される。 $H_0(z)$ は 5 タップで、 $H_1(z)$ は 3 タップであることがわかる。

10

【 0 0 6 2 】

$$H_0(z) = (-1 + 2z^{-1} + 6z^{-2} + 2z^{-3} - z^{-4}) / 8 \quad \cdots (1)$$

$$H_1(z) = (-1 + 2z^{-1} - z^{-2}) / 2 \quad \cdots (2)$$

【 0 0 6 3 】

これら式 (1) および式 (2) によれば、低域成分および高域成分の係数を、直接的に算出することができる。このようなフィルタを用いる分析フィルタ処理における演算方法で最も一般的な演算方法は、畳み込み演算と呼ばれる方法である。この畳み込み演算は、デジタルフィルタの最も基本的な実現手段であり、フィルタのタップ係数に実際の入力データを畳み込み乗算するものである。しかしながら、この畳み込み演算ではタップ長に応じて計算負荷が増大するので、論文「W.Swelden, "The lifting scheme :A custom-design construction of Biorthogonal wavelets.", Appl. Comput. Harmon. Anal., vol3, no.2, pp.186-200, 1996」で紹介されたリフティング技術を用いてフィルタ処理の演算を行う方法がある。

20

【 0 0 6 4 】

図 5 は、ウェーブレット変換処理およびウェーブレット逆変換処理によるデータの変化の様子を示す模式図である。図 5 A は、ウェーブレット変換されるベースバンドの画像データを模式的に示し、図 5 B は、図 5 A の画像データが分割レベル 3 でウェーブレット変換されて得られる係数データを模式的に示し、図 5 C は、図 5 B の係数データがウェーブレット逆変換されて得られるベースバンドの画像データを模式的に示している。

30

【 0 0 6 5 】

図 5 において、各模様で示される領域はそれぞれプレシントを示している。また、図 5 A 乃至図 5 C において同じ模様のプレシントは、互いに対応するプレシントを示している。例えば、図 5 A において右上左下の斜め線で示されるベースバンドの画像データがウェーブレット変換されると、図 5 B において右上左下の斜め線で示される、帯域毎に階層的に分割された係数データが生成され、さらにそれらの係数データがウェーブレット逆変換されると、図 5 C において右上左下の斜め線で示されるベースバンドの画像データが生成される。なお、説明の簡略化のため、図 5 においては、1 ピクチャのライン数を 31 ラインとする。

【 0 0 6 6 】

40

詳細については後述するが、図 5 A に示されるように、リフティング技術を用いてフィルタ処理の演算を行う場合、一般的に、ピクチャの一番上のプレシントは、上から 2 番目以降のプレシントよりも多くのライン数を必要とする。また、図 5 B に示されるように、リフティング技術を用いてフィルタ処理の演算を行う場合、一般的に、ピクチャの一番上のプレシントと一番下のプレシントは、各分割レベルにおける係数データのライン数が他のプレシントと異なる。

【 0 0 6 7 】

さらに、リフティング技術を用いてフィルタ処理の演算を行う場合、図 5 A および図 5 C に示されるように、あるプレシントについて、ベースバンドの画像データをウェーブレット変換し、さらにウェーブレット逆変換すると、変換前のライン番号と異なるライン

50

番号のベースバンドの画像データが復元される。つまり、ウェーブレット変換時のプレシ
ンクトと、そのプレシンクトに対応するウェーブレット逆変換時のプレシンクトとでは、
ベースバンドの画像データのライン番号が互いに異なる。

【 0 0 6 8 】

また、ピクチャの一番上のプレシンクトの場合、ウェーブレット変換前のベースバンド
の画像データと、ウェーブレット逆変換後に得られるベースバンドの画像データとでライ
ン数が異なる。従って、図 5 A および図 5 C に示されるように、ベースバンドの画像デー
タをウェーブレット変換し、さらにウェーブレット逆変換するとライン数が減少する。従
って、図 5 C に示されるように 3 1 ラインの画像データを復元するためには、図 5 A に示
されるように 3 1 ラインより多くのラインの画像データをウェーブレット変換する必要が
ある。つまり、ウェーブレット変換時にはベースバンドの画像データにダミーデータが付
加される。なお、図 5 C に示されるピクチャの一番下のプレシンクトの画像データを復元
するために、ウェーブレット逆変換処理においてもダミーデータが利用される。

10

【 0 0 6 9 】

次に、リフティング技術について説明する。

【 0 0 7 0 】

図 6 は、 5×3 フィルタのリフティングを用いた例を示したものである。以下、このフ
ィルタリング方法について説明する。

【 0 0 7 1 】

図 6 において、最上段部、中段部および最下段部は、それぞれ入力画像の画素列、高域
成分出力および低域成分出力を示す。最上段は、入力画像の画素列に限らず、先のフィル
タ処理で得られた係数でもよい。ここでは、最上段部が入力画像で画素列であるものとし
、四角印 () が偶数番目 (最初を 0 番目とする) の画素またはライン、丸印 () が奇
数番目の画素またはラインとする。

20

【 0 0 7 2 】

先ず第 1 段階として、次式 (3) により入力画素列から高域成分の係数 d_i^1 を生成する
。

【 0 0 7 3 】

$$d_i^1 = d_i^0 - 1/2 (s_i^0 + s_{i+1}^0) \quad \cdots (3)$$

【 0 0 7 4 】

30

次に第 2 段階として、この生成された高域成分の係数と、入力画像の奇数番目の画素と
を用いて、次式 (4) により低域成分の係数 s_i^1 を生成する。

【 0 0 7 5 】

$$s_i^1 = s_i^0 + 1/4 (d_{i-1}^1 + d_i^1) \quad \cdots (4)$$

【 0 0 7 6 】

分析フィルタ側では、このようにして、フィルタリング処理により入力画像の画素デー
タを低域成分と高域成分とに分解する。

【 0 0 7 7 】

次に、このようなウェーブレット変換により生成された係数を復元するウェーブレット
逆変換を行う合成フィルタ側の処理について、図 7 を参照して概略的に説明する。この図
7 は、上述の図 6 と対応し、 5×3 フィルタを用い、リフティング技術を適用した例を示
す。図 7 において、最上段部は、ウェーブレット変換により生成された入力係数を示し、
丸印 () が高域成分の係数、四角印 () が低域成分の係数をそれぞれ示す。

40

【 0 0 7 8 】

先ず第 1 段階として、次式 (5) に従い、入力された低域成分および高域成分の係数か
ら、偶数番目 (最初を 0 番目とする) の係数 s_i^0 が生成される。

【 0 0 7 9 】

$$s_i^0 = s_i^1 - 1/4 (d_{i-1}^1 + d_i^1) \quad \cdots (5)$$

【 0 0 8 0 】

次に第 2 段階として、次式 (6) に従い、上述の第 1 段階で生成された偶数番目の係数

50

s_i^0 と、入力された高域成分の係数 d_i^1 から、奇数番目の係数 d_i^0 が生成される。

【 0 0 8 1 】

$$d_i^0 = d_i^1 + 1 / 2 (s_i^0 + s_{i+1}^0) \quad \cdots (6)$$

【 0 0 8 2 】

合成フィルタ側では、このようにして、フィルタリング処理により低域成分および高域成分の係数を合成し、ウェーブレット逆変換を行う。

【 0 0 8 3 】

図 8 は、 5×3 フィルタのリフティングによるフィルタリングを分解レベル 3 まで実行したときの分析フィルタ処理および合成フィルタ処理の様子の例を示す模式図である。図 8 の左側は分析フィルタ処理におけるリフティング演算の様子の例を示しており、図 8 の右側は合成フィルタ処理におけるリフティング演算の様子の例を示している。

【 0 0 8 4 】

より具体的に説明すると、図 8 の丸や四角は、画像データまたは係数データを示しており、それらを結ぶ直線は、リフティング演算における画像データや係数データの関係を示している。つまり、図 8 においては、丸や四角の 3 列毎に 1 つのリフティング演算が示されており、各 3 列のうち、最も左の列の丸や四角がリフティング演算の入力データを示し、中央の列および最も右の列の丸や四角がリフティング演算の出力データを示している。各入力データがリフティング演算されることにより、その入力データと直線で結ばれる出力データが生成される。換言すれば、各出力データは、その出力データと直線で結ばれる入力データを用いて生成される。

【 0 0 8 5 】

図 8 の中央縦方向の点線の左側において、左から 1 列目乃至 3 列目の丸、四角、およびそれらを結ぶ直線は分割レベル 1 の分析フィルタ処理のリフティング演算の様子を示し、左から 4 列目乃至 6 列目の丸、四角、およびそれらを結ぶ直線は分割レベル 2 の分析フィルタ処理のリフティング演算の様子を示し、左から 7 列目乃至 9 列目の丸、四角、およびそれらを結ぶ直線は分割レベル 3 の分析フィルタ処理のリフティング演算の様子を示している。また、図 8 の中央縦方向の点線の右側において、左から 1 列目乃至 3 列目の丸、四角、およびそれらを結ぶ直線は分割レベル 3 の合成フィルタ処理のリフティング演算の様子を示し、左から 4 列目乃至 6 列目の丸、四角、およびそれらを結ぶ直線は分割レベル 2 の合成フィルタ処理のリフティング演算の様子を示し、左から 7 列目乃至 9 列目の丸、四角、およびそれらを結ぶ直線は分割レベル 1 の合成フィルタ処理のリフティング演算の様子を示している。

【 0 0 8 6 】

図 8 において、左上右下の斜め線模様で示される丸は、ベースバンドの画像データの、ライン番号が奇数のライン（奇数ライン）を示し、左上右下の斜め線模様で示される四角は、ベースバンドの画像データの、ライン番号が偶数のライン（偶数ライン）を示している。なお、ライン番号は、画像の一番上のラインを「0」とし、上から下に向かう順にインクリメントされる。

【 0 0 8 7 】

また図 8 において、左下右上の斜め線模様で示される丸と四角は、それぞれ、リフティング演算の途中の計算に利用される途中計算用の奇数番目の係数と偶数番目の係数を示しており、図 8 において黒丸（ ）と黒四角（ ）は、それぞれ、リフティング演算結果として得られる高域成分と低域成分（周波数成分）を示している。

【 0 0 8 8 】

なお、図 8 では、説明の簡略化のため、垂直方向についてのフィルタ処理のみ説明しており、水平方向の分析フィルタ処理および合成フィルタ処理については省略している。

【 0 0 8 9 】

より具体的に説明する。図 8 の中央縦方向の点線の左側において、左から 1 列目に示される画像データが入力されると、ウェーブレット変換部 11 は、それらの画像データに対して分割レベル 1 の分析フィルタ処理（リフティング演算）を行い、左から 2 列目に示さ

れる分割レベル1の高域成分と、左から3列目に示される分割レベル1の低域成分を生成する。この左から2列目に示される分割レベル1の高域成分は、これ以上分析フィルタ処理が行われず、係数並び替え用バッファ部13に供給されて保持される。また、左から3列目に示される分割レベル1の低域成分は、再度分析フィルタ処理を行うために、途中計算用データとして途中計算用バッファ部12に供給されて保持される。

【0090】

次に、ウェーブレット変換部11は、途中計算用バッファ部12に保持させた途中計算用データに対して、図8の中央縦方向の点線の左側において、左から4列目乃至6列目に示されるように、分割レベル2の分析フィルタ処理（リフティング演算）を行う。左から4列目の丸および四角は、それぞれ、左から3列目の同じ行の四角と同じ係数データを示してあり、丸が奇数ラインの係数データを示し、四角が偶数ラインの係数データを示している。

10

【0091】

ウェーブレット変換部11は、左から4列目に示される係数データに対して分割レベル2の分析フィルタ処理（リフティング演算）を行い、左から5列目に示される分割レベル2の高域成分と、左から6列目に示される分割レベル2の低域成分を生成する。この左から5列目に示される分割レベル2の高域成分は、これ以上分析フィルタ処理が行われず、係数並び替え用バッファ部13に供給されて保持される。また、左から6列目に示される分割レベル2の低域成分は、再度分析フィルタ処理を行うために、途中計算用データとして途中計算用バッファ部12に供給されて保持される。

20

【0092】

次に、ウェーブレット変換部11は、途中計算用バッファ部12に保持させた途中計算用データに対して、図8の中央縦方向の点線の左側において、左から7列目乃至9列目に示されるように、分割レベル3の分析フィルタ処理（リフティング演算）を行う。左から7列目の丸および四角は、それぞれ、左から6列目の同じ行の四角と同じ係数データを示してあり、丸が奇数ラインの係数データを示し、四角が偶数ラインの係数データを示している。

【0093】

ウェーブレット変換部11は、左から7列目に示される係数データに対して分割レベル3の分析フィルタ処理（リフティング演算）を行い、左から8列目に示される分割レベル3の高域成分と、左から9列目に示される分割レベル3の低域成分を生成する。分割レベル3は、予め最上位レベルとして設定されており、これ以上分析フィルタ処理は行われない。従って、左から8列目に示される分割レベル3の高域成分も、左から9列目に示される分割レベル3の低域成分も、係数並び替え用バッファ部13に供給されて保持される。

30

【0094】

つまり、図8の中央縦方向の点線の左側において、左から3列目および4列目の係数データを囲む点線の角丸四角、並びに、左から5列目および6列目の係数データを囲む点線の角丸四角は、途中計算用バッファ部12を示してあり、左から2列目、5列目、8列目、および9列目の係数データをそれぞれ囲む実線の角丸四角は、係数並び替え用バッファ部13を示している。

40

【0095】

以上のように、ウェーブレット変換部11は、ウェーブレット変換処理により、分割レベル1から分割レベル3に向かう順にリフティング演算を行うことにより、高域から低域に向かう順に各分割レベルの係数データを生成する。ただし、詳細については後述するが、実際には、画像データは1ラインずつ入力されるので、ウェーブレット変換部11は、画像データが2ライン入力される毎にリフティング演算を行うことができる。つまり、入力された画像データや、係数並び替え用バッファ部13に保持させる係数データも、次のリフティング演算に利用する場合がある。従って、そのような画像データは、入力されたときのリフティング演算に使用されるだけでなく、途中計算用データとして途中計算用バ

50

ッファ部 1 2 に保持され、次のラインが入力されたときに行われるリフティング演算にも利用される。係数データも同様に、係数並び替え用バッファ部 1 3 に保持されるだけでなく、途中計算用データとして途中計算用バッファ部 1 2 に保持され、後に行われるリフティング演算にも利用される。

【 0 0 9 6 】

ウェーブレット変換部 1 1 は、上述したリフティング演算により、リフティング演算結果の係数データ（黒丸（ ）と黒四角（ ））を、それぞれの左側に付した番号順に生成する。

【 0 0 9 7 】

これに対して、図 8 中央縦方向の点線の右側には、ウェーブレット逆変換部 2 3 による合成フィルタ処理におけるリフティング演算の様子が示される。

10

【 0 0 9 8 】

図 8 の中央縦方向の点線の右側において、左から 1 列目に示される係数データ（黒丸（ ）と黒四角（ ））は、それぞれ、自分自身と同じ行の、図 8 の中央縦方向の点線の左側の左から 8 列目または 9 列目の係数データと同じデータである。つまり、ウェーブレット逆変換部 2 3 は、図 8 の中央縦方向の点線の右側において左から 1 列目に示される分割レベル 3 の低域成分（黒四角（ ））と高域成分（黒丸（ ））を合成する、分割レベル 3 の合成フィルタ処理（リフティング演算）を行い、左から 2 列目に示される分割レベル 2 の低域成分の偶数ラインの係数データと、左から 3 列目に示される分割レベル 2 の低域成分の奇数ラインの係数データを生成する。

20

【 0 0 9 9 】

この左から 2 列目および 3 列目に示される分割レベル 2 の低域成分と、新たに供給された分割レベル 2 の高域成分に対して、ウェーブレット逆変換部 2 3 は、再度リフティング演算（分割レベル 2 の合成フィルタ処理）を行う（左から 4 列目乃至 6 列目）。左から 4 列目の左下右上の斜線模様の四角は、それぞれ自分自身と同じ行の、左から 2 列目の四角または左から 3 列目の丸と同じ係数データを示している。また、左から 4 列目の黒丸（ ）は、それぞれ自分自身と同じ行の、図 8 の中央縦方向の点線の左側において左から 5 列目の黒丸（ ）と同じ係数データ、すなわち、新たに供給された分割レベル 2 の高域成分を示している。

【 0 1 0 0 】

30

ウェーブレット逆変換部 2 3 は、図 8 の中央縦方向の点線の右側において左から 4 列目に示される分割レベル 2 の低域成分と高域成分を合成する、分割レベル 2 の合成フィルタ処理（リフティング演算）を行い、左から 5 列目に示される分割レベル 1 の低域成分の偶数ラインの係数データと、左から 6 列目に示される分割レベル 1 の低域成分の奇数ラインの係数データを生成する。

【 0 1 0 1 】

この左から 5 列目および 6 列目に示される分割レベル 1 の低域成分と、新たに供給された分割レベル 1 の高域成分に対して、ウェーブレット逆変換部 2 3 は、再度リフティング演算（分割レベル 1 の合成フィルタ処理）を行う（左から 7 列目乃至 9 列目）。左から 7 列目の左下右上の斜線模様の四角は、それぞれ自分自身と同じ行の、左から 5 列目の四角または左から 6 列目の丸と同じ係数データを示している。また、左から 7 列目の黒丸（ ）は、それぞれ自分自身と同じ行の、図 8 の中央縦方向の点線の左側において左から 2 列目の黒丸（ ）と同じ係数データ、すなわち、新たに供給された分割レベル 1 の高域成分を示している。

40

【 0 1 0 2 】

ウェーブレット逆変換部 2 3 は、図 8 の中央縦方向の点線の右側において左から 7 列目に示される分割レベル 1 の低域成分と高域成分を合成する、分割レベル 1 の合成フィルタ処理（リフティング演算）を行い、左から 8 列目に示されるベースバンドの画像データの偶数ラインと、左から 9 列目に示されるベースバンドの画像データの奇数ラインを生成する。

50

【 0 1 0 3 】

以上のような分析フィルタ処理および合成フィルタ処理のリフティング演算の関係から、プレシントは、曲線 5 1 および曲線 5 2 に示されるように区切られる。つまり、ウェーブレット変換時においては、ピクチャの一番上のプレシントである第 1 プレシントの画像データは、ライン番号 0 乃至ライン番号 1 4 の 1 5 ラインが必要であり、ピクチャの上から 2 番目のプレシントである第 2 プレシントの画像データは、ライン番号 1 5 乃至ライン番号 2 2 の 8 ラインが必要である。なお図 8 に示されるように実際には、第 2 プレシントの分析フィルタ処理において、第 1 プレシントの画像データや係数データも利用される。

【 0 1 0 4 】

また、ウェーブレット逆変換時においては、第 1 プレシントの係数データより、ライン番号 0 の 1 ラインの画像データが生成され、第 2 プレシントの係数データより、ライン番号 1 乃至ライン番号 8 の 8 ラインの画像データが生成される。なお図 8 に示されるように実際には、第 2 プレシントの合成フィルタ処理において、第 1 プレシントの画像データや係数データも利用される。

【 0 1 0 5 】

図 8 の中央縦方向の点線の右側において、黒丸 () および黒四角 () の左側に付した番号は、それぞれの係数データが合成フィルタ処理において利用される順序を示している。また、その数字の横に示される括弧内の数字は、その係数データが分析フィルタ処理において生成された順序を示している。つまり、図 8 の中央縦方向の点線の右側において示される括弧内の数字は、図 8 の中央縦方向の点線の左側に示される数字と対応する。

【 0 1 0 6 】

つまり、分析フィルタにおける係数データの生成順と、合成フィルタにおける係数データの利用順は互いに異なる。従って図 8 の中央において白抜き矢印で示されるように、分析フィルタ処理において生成された係数データは、合成フィルタ処理が行われる前に、その順序を並び替えられる必要がある。

【 0 1 0 7 】

係数並び替え部 1 4 は、係数並び替え用バッファ部 1 3 に保持された係数データを、合成フィルタ処理における利用順に読み出すことにより、この係数データの並び替えを行う。つまり、エントロピ符号化部 1 5 は、合成フィルタ処理における利用順に係数データを符号化する。

【 0 1 0 8 】

符号化部 1 0 および復号部 2 0 の各部は、それぞれの処理を、図 9 に模式的に示されるように、互いに並列的に実行する。つまり、符号化部 1 0 による、画像データの入力、ウェーブレット変換 (DWT)、係数の並び替え、およびエントロピ符号化 (VLC) の各処理と、復号部 2 0 による、エントロピ復号 (VLD)、ウェーブレット逆変換 (IDWT)、および画像データ出力の各処理は、図 9 の矢印に示されるように、互いに並行して実行される。各矢印は 1 プレシントに対する処理を示しており、各矢印の下に付された数字は、その矢印において処理されるライン数を示している。例えば、各処理において一番左の矢印で示される第 1 プレシントの場合、1 5 ラインの画像データが入力され、4 ラインの係数データにウェーブレット変換されて符号化され、さらに復号された後ウェーブレット逆変換されて 1 ラインの画像データとして出力される。これは図 8 を参照して行った説明と一致する。

【 0 1 0 9 】

このように並行して処理を行うことにより、入力から出力までの遅延時間を、図 9 において白抜き矢印に示されるように、1 5 ライン分の時間と各処理の処理時間の合計にすることができる。これに対して例えば 1 ピクチャ全体を対象としてウェーブレット変換する場合、1 ピクチャ分以上の遅延時間が発生する。従って、プレシント単位で各処理を行い、かつ、図 9 に示されるように各処理を互いに並行して実行させることにより、入力から出力までの遅延時間を大幅に低減させることができる。

【 0 1 1 0 】

しかしながら、ソフトウェアエンコーダやソフトウェアデコーダの場合、図 1 0 に示されるようなパーソナルコンピュータにおいてソフトウェアプログラムが実行されることにより実現される。

【 0 1 1 1 】

図 1 0 は、パーソナルコンピュータの構成例を示すブロック図である。

【 0 1 1 2 】

図 1 0 において、パーソナルコンピュータ 1 0 0 の CPU 1 0 1 は、ソフトウェアプログラムを実行することにより、各種の処理を実行する演算処理部である。CPU 1 0 1 は、共有バスであるバス 1 0 4 を介して ROM (Read Only Memory) 1 0 2 および RAM (Random Access Memory) 1 0 3 と相互に接続されている。ROM 1 0 2 には予めソフトウェアプログラムやデータが格納されている。RAM 1 0 3 には、ROM 1 0 2 や記憶部 1 1 3 に格納されているソフトウェアプログラムやデータがロードされる。RAM 1 0 3 にはまた、CPU 1 0 1 が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。

10

【 0 1 1 3 】

CPU 1 0 1、ROM 1 0 2、および RAM 1 0 3 は、バス 1 0 4 を介して相互に接続されている。このバス 1 0 4 にはまた、入出力インタフェース 1 1 0 も接続されている。

【 0 1 1 4 】

入出力インタフェース 1 1 0 には、キーボード、マウスなどよりなる入力部 1 1 1、CRT (Cathode Ray Tube)、LCD (Liquid Crystal Display) などよりなるディスプレイ、並びにスピーカなどよりなる出力部 1 1 2、ハードディスクなどより構成される記憶部 1 1 3、モデムなどより構成される通信部 1 1 4 が接続されている。通信部 1 1 4 は、例えばインターネットに代表されるネットワークを介しての通信処理を行う。

20

【 0 1 1 5 】

入出力インタフェース 1 1 0 にはまた、必要に応じてドライブ 1 1 5 が接続され、磁気ディスク、光ディスク、光磁気ディスク、或いは半導体メモリなどのリムーバブルメディア 1 2 1 が適宜装着され、それらから読み出されたコンピュータプログラムが、必要に応じて記憶部 1 1 3 にインストールされる。

【 0 1 1 6 】

なお、図示は省略するが、CPU 1 0 1 の内部には、キャッシュメモリが設けられている。このキャッシュメモリは、RAM 1 0 3 に比べて高速に動作するものの、RAM 1 0 3 に比べて容量が小さい。つまり、このキャッシュメモリには、一度に格納可能なデータ量が制限される。従って、CPU 1 0 1 は、使用頻度の高いデータや、近いうちを使用する予定のあるデータをこのキャッシュメモリに格納し、使用頻度が少ないデータや当面利用する予定の無いデータ等を RAM 1 0 3 に保持させる。

30

【 0 1 1 7 】

図 1 0 に示されるように、パーソナルコンピュータ 1 0 0 には、演算処理部は CPU 1 0 1 の 1 つしか存在しない。従って、図 1 に示される符号化部 1 0 のようなソフトウェアエンコーダの各処理を図 9 に示されるように、実際に並列に実行させることはできない。そのため、各処理が擬似的に並列に実行されているようにするためには、CPU 1 0 1 を時分割して各処理に割り当てる必要がある。その場合、時分割の単位として、プレシントを用いる方法が考えられる。

40

【 0 1 1 8 】

つまり、図 1 に示されるように、供給される画像データに対して、1 プレシントを対象としてウェーブレット変換処理を行い、係数を並び替え、エントロピ符号化処理を施すようにする。これをプレシント毎に複数回繰り返すことにより、1 ピクチャの画像データを処理するようにする。このようにすることにより、ピクチャ全体を対象としてウェーブレット変換する場合よりも、ウェーブレット変換処理の負荷や遅延時間を低減させることができる。

【 0 1 1 9 】

50

しかしながら、単純に1プレシント全体を対象としてウェーブレット変換処理を行う場合、入力される画像データを1プレシント分蓄積しなければならず、メモリ量が増大し、回路規模やコストが増大するだけでなく、負荷や遅延時間も増大してしまう恐れがある。

【0120】

そこで、上述したようにウェーブレット変換部11がリフティング演算を用いてウェーブレット変換処理を行うことに着目し、1プレシントの画像データに対して行うウェーブレット変換処理を効率よく行うことにより、この画像データの蓄積量を低減させるようにする。

【0121】

図11にウェーブレット変換処理におけるリフティング演算の手順の例を示す。図11においては、ピクチャの上から2番目のプレシントである第2プレシントを例に説明する。第1プレシントや第3プレシント以降のプレシントの場合も基本的には同様である。

【0122】

図11は、基本的に図8と同様の図であり、分析フィルタ処理として 5×3 フィルタを用いたときのリフティング演算の様子を示す図である。従って図11の詳細についての説明は省略する。なお、黒丸()と黒四角()の右横または下に付された番号は、各係数データの合成フィルタ処理される順序を示している。

【0123】

図11において、曲線151により囲まれる部分が符号化部10における第2プレシント(第2プレシントに対するリフティング演算の様子)を示している。図8を参照して説明したように、リフティング演算は、白抜き矢印で示されるように、図中左から右へ、すなわち、より高域の周波数成分を生成するリフティング演算をより先に実行するように進む。従って、第2プレシントに対するリフティング演算は、画像データが2ライン入力される度に、曲線161乃至曲線164に示されるように行うことができる。つまり、ウェーブレット変換部11は、ライン番号15および16の画像データが入力されると、曲線161に囲まれる部分のリフティング演算を行い、19番目に合成フィルタ処理される係数データを生成し、ライン番号17および18の画像データが入力されると、曲線162に囲まれる部分のリフティング演算を行い、20番目に合成フィルタ処理される係数データを生成し、さらに、15番目に合成フィルタ処理される係数データを生成する。

【0124】

続いて、ウェーブレット変換部11は、ライン番号19および20の画像データが入力されると、曲線163に囲まれる部分のリフティング演算を行い、24番目に合成フィルタ処理される係数データを生成し、ライン番号21および22の画像データが入力されると、曲線164に囲まれる部分のリフティング演算を行い、25番目に合成フィルタ処理される係数データを生成し、さらに、18番目に合成フィルタ処理される係数データを生成し、最後に、6番目に合成フィルタ処理される係数データと5番目に合成フィルタ処理される係数データを生成する。

【0125】

このように、2ライン画像データが入力される毎に、実行可能なリフティング演算を順次行うことにより、ウェーブレット変換部11は、1プレシント分の画像データを蓄積する必要がなくなり、必要なメモリ容量を低減させ、回路規模や製造コストを低減させることができるだけでなく、ウェーブレット変換処理の負荷や遅延時間を低減させることができる。

【0126】

なお、この符号化部10における第2プレシントに対応する、復号部20における第2プレシントは、図11の曲線152により囲まれる部分に示されており、係数データに付された番号で示されるように、合成フィルタ処理は、白抜き矢印で示されるように、図中右から左に向かう向き、すなわち、分析フィルタ処理により帯域毎に分割された各周

10

20

30

40

50

波数成分のうち、より低域の周波数成分であり、かつ、同じ帯域の中では、より先に生成された周波数成分を、より先に処理するように進められる。符号化部 10 において、エントロピ符号化の処理順は、上述したように、ウェーブレット逆変換（合成フィルタ処理）の処理順と同じである。つまり、このエントロピ符号化も、分析フィルタ処理により帯域毎に分割された各周波数成分のうち、より低域の周波数成分であり、かつ、同じ帯域の中では、より先に生成された周波数成分を、より先に処理するように進められる。

【0127】

黒丸や黒四角で示される、ウェーブレット変換部 11 より出力される係数データは、係数並び替え部 14 により、その番号順にエントロピ符号化部 15 に供給される。エントロピ符号化部 15 は、その供給順に各係数データに対してエントロピ符号化を行い、それぞれの符号化データを得る。つまり、エントロピ符号化部 15 は、ウェーブレット変換処理により、図 11 において点線の楕円で囲まれた、5 番目に合成フィルタ処理される係数データと 6 番目に合成フィルタ処理される係数データが生成されるまで、復号部 20 における第 2 プレシントの係数データに対してエントロピ符号化を開始することができない。

【0128】

従って、CPU 101 においては、図 12 に示されるように符号化部 10 の各処理が進められる。図 12 において、CPU 101 は、ソフトウェアプログラムを実行することにより符号化部 10 の機能を実現する。なお、その CPU 101 に対して符号化される画像データが、通常再生の速度、つまりビデオ信号の同期タイミングに合わせて入力される。つまり、CPU 101 には画像データが 1 ラインずつ、ビデオ信号の水平同期タイミングに合わせて略一定の間隔で供給される。

【0129】

CPU 101 は、符号化部 10 における第 2 プレシントについて、画像データが 2 ライン入力される毎に実行可能なリフティング演算（分析フィルタ処理）を行う（DWT（Lev 1）乃至 DWT（Lev 3））。

【0130】

図 12 において、「DWT（Lev 1）」と記述された四角は、CPU 101 に分割レベル 1 の合成フィルタ処理が割り当てられる時間の例を示しており、「DWT（Lev 2）」と記述された四角は、CPU 101 に分割レベル 2 の合成フィルタ処理が割り当てられる時間の例を示しており、「DWT（Lev 3）」と記述された四角は、CPU 101 に分割レベル 3 の合成フィルタ処理が割り当てられる時間の例を示している。つまり、図中、一番上の段の四角（DWT（Lev 1））に示される時間（ライン番号 16（および 15）の画像データが入力されてから、ライン番号 18（および 17）の画像データが入力されるまでの間）において、CPU 101 が図 11 の曲線 161 に囲まれた部分のリフティング演算を行い、上から 2 段目の四角（DWT（Lev 1））と 3 段目の四角（DWT（Lev 2））に示される時間（ライン番号 18（および 17）の画像データが入力されてから、ライン番号 20（および 19）の画像データが入力されるまでの間）において、CPU 101 が図 11 の曲線 162 に囲まれた部分のリフティング演算を行い、上から 4 段目の四角（DWT（Lev 1））に示される時間（ライン番号 20（および 19）の画像データが入力されてから、ライン番号 22（および 21）の画像データが入力されるまでの間）において、CPU 101 が図 11 の曲線 163 に囲まれた部分のリフティング演算を行い、上から 5 段目の四角（DWT（Lev 1））と 6 段目の四角（DWT（Lev 2））と 7 段目の四角（DWT（Lev 3））に示される時間（ライン番号 22（および 21）の画像データが入力した後の時間）において、CPU 101 が図 11 の曲線 164 に囲まれた部分のリフティング演算を行うことが示されている。

【0131】

CPU 101 は、各リフティング演算において、符号化する係数データを生成すると、それを順次係数並び替え用バッファ部 13 に保持させる。この係数並び替え用バッファ部 13 は、例えば、CPU 101 内のキャッシュメモリや RAM 103 により構成される。

【0132】

そして、5 番目に合成フィルタ処理される係数データと 6 番目に合成フィルタ処理され

10

20

30

40

50

る係数データを生成すると、CPU 1 0 1 は、復号部 2 0 における第 2 プレシントの係数データに対してエントロピ符号化を開始する。なお、このとき最後に生成された、5 番目に合成フィルタ処理される係数データと 6 番目に合成フィルタ処理される係数データは、データ量も少なく、かつ直ぐに使用するので、CPU 1 0 1 において保持したまま、係数並び替え用バッファ部 1 3 に保持させずに、エントロピ符号化処理を開始する。

【 0 1 3 3 】

このように、最後に生成された係数データを最初に符号化するという、係数並び替えの特徴を利用して、5 番目に合成フィルタ処理される係数データと 6 番目に合成フィルタ処理される係数データの、係数並び替え用バッファ部 1 3 への保持を省略することにより、係数並び替え用バッファ部 1 3 として必要なメモリ容量を低減させることができる。これにより、回路規模および製造コストを低減させることができるだけでなく、係数並び替え用バッファ部 1 3 への書き込みおよび読み出しの回数が低減されるので、ウェーブレット変換処理および符号化の負荷や遅延時間を低減させることができる。

【 0 1 3 4 】

エントロピ符号化を開始した CPU 1 0 1 は、5 番目乃至 1 2 番目に合成フィルタ処理される係数データをその番号順に 1 ラインずつ符号化し、得られた符号化データを順次出力する。

【 0 1 3 5 】

図 1 2 において、「VLC 5」と記述された四角は、CPU 1 0 1 に、5 番目に合成フィルタ処理される係数データを符号化するエントロピ符号化処理が割り当てられる時間の例を示している。「VLC 6」乃至「VLC 1 2」も同様に、6 番目乃至 1 2 番目に合成フィルタ処理される係数データを符号化するエントロピ符号化処理が、それぞれ割り当てられる時間の例を示している。

【 0 1 3 6 】

なお、図 1 2 において、図中縦方向上から下に向かう向きに時間の流れを示しているが、その縦方向の長さ（スケール）が、時間の長さを正確に示しているわけではない。例えば、図 1 2 において、画像データの入力 が 2 ラインずつ入力され、さらに、各入力の間隔が一定でないように示されているが、実際には、画像データはビデオ信号の水平同期タイミングに合わせた略一定の周期で 1 ラインずつ入力される。

【 0 1 3 7 】

また、上述したように、図 1 2 に示される各四角（係数並び替え用バッファ部 1 3 を除く）は、リフティング演算やエントロピ符号化等の処理が CPU 1 0 1 に割り当てられる時間を示しているが、これは、CPU 1 0 1 がその四角に示される時間において割り当てられた処理を行うことを示すものであり、CPU 1 0 1 による各処理の処理時間を示すものではない。つまり、CPU 1 0 1 が四角に示される時間のうち一部の時間で処理を終了させる場合もある。また、上述したように図 1 2 において縦方向の長さが時間の長さを示しているわけではなく、四角の縦方向の長さも処理の割り当て時間を正確に示すものではない。

【 0 1 3 8 】

図 1 2 のような処理の流れの場合、ウェーブレット変換処理（DWT）を行っている期間中に入力される画像データに対しては、CPU 1 0 1 は、分析フィルタ処理をリアルタイム（即時的）に行うことができるが、エントロピ符号化処理（VLC）を行っている期間に入力される入力される画像データに対しては、分析フィルタ処理をリアルタイム（即時的）に行うことができない。

【 0 1 3 9 】

つまり、図 1 3 に示されるように、通常の再生速度のビデオ信号のように、所定の速度で入力される画像データは、そのビデオ信号の垂直同期信号（vsync）の同期タイミングに同期してピクチャ毎に入力される。各ピクチャの画像データは、ビデオ信号の水平同期信号（hsync）の同期タイミングに同期してライン毎に入力される。つまり、画像データは略定期的に（略一定のペースで）1 ラインずつ入力される。これに対して、CPU 1 0 1 は、図 1 3 に示されるようにプレシント毎にウェーブレット変換処理（DWT）とエント

10

20

30

40

50

ロピ符号化処理（VLC）を交互に実行する。

【0140】

従って、例えば、図13において点線の楕円171に囲まれる矢印のように、CPU101が第1プレシントの係数データを符号化中（P1VLC）に発生する水平同期タイミングに合わせて入力される画像データに対して、CPU101はリアルタイム（即時的）にウェーブレット変換処理（DWT）を行うことができない（符号化（VLC）中なので）。そこで、その画像データについては、CPU101が次の第2プレシントに対するウェーブレット変換処理（P2DWT）を開始するまでバッファする（保持しておく）必要がある。

【0141】

仮に、水平同期タイミング間に1プレシント分の符号化処理を全て終了させることができるのであれば、CPU101は、画像データのバッファリングを不要とし、リアルタイムにウェーブレット変換処理を行うことができる。しかしながら、このような方法の場合、CPU101にかかる負荷が極めて大きく、このような方法を実現するためには、CPU101の動作周波数を非常に高くする必要があり、コストが増大してしまう恐れがあるので現実的ではない。現実的なコストで調達可能なCPUの処理速度で画像データを処理するためには、上述したように入力画像データのバッファリングが必要になる。

【0142】

なお、図13は、画像データ入力とCPU101において実行される処理のタイミングの関係を模式的に示す図であり、図13に示される水平同期信号（hsync）の矢印は、説明のための代表的な水平同期タイミングを概略的に示したものであり、全ての水平同期タイミングを正確に示すものではない。また、図13においても、図12の場合と同様に、縦方向上から下向きに時間の流れが示されているが、その縦方向の長さが時間の長さを正確に示しているわけではない。

【0143】

次に、このような流れで処理を実行する符号化部10の実現化について、より具体的に説明する。図14は、上述した流れで処理を実行する符号化部10を実現するソフトウェアプログラムを実行するCPU101が有する機能を示す機能ブロック図である。

【0144】

図14において、CPU101は、制御部201、分析フィルタ処理実行部211、係数並び替え処理実行部212、エントロピ符号化処理実行部213、途中計算用データ読み出し部214、途中計算用データ書き込み部215、出力係数書き込み部216、および出力係数読み出し部217を有する。

【0145】

制御部201は、分析フィルタ処理実行部211乃至出力係数読み出し部217の各部の動作を制御することにより、符号化部10の各処理の流れを制御する。また、制御部201は、後述する画像データバッファ部221のデータ入出力も制御する。

【0146】

分析フィルタ処理実行部211は、制御部201に制御されて、入力される画像データに対して分析フィルタ処理（リフティング演算）を行う、図1のウェーブレット変換部11の機能を実現する分析フィルタ処理を実行する。その分析フィルタ処理を実行する際、分析フィルタ処理実行部211は、次のリフティング演算に必要な途中計算用データ（画像データや係数データ）を、途中計算用データ読み出し部214を介して途中計算用バッファ部12より取得する。また、分析フィルタ処理実行部211は、リフティング演算により得られた途中計算用データを、途中計算用データ書き込み部215を介して途中計算用バッファ部12に供給し、保持させる。さらに、分析フィルタ処理実行部211は、リフティング演算により得られた、分析フィルタ処理をこれ以上行わない係数データ（最上位の分割レベルの低域成分および高域成分、並びに、最上位以外の分割レベルの高域成分）を、出力係数として、出力係数書き込み部216を介して係数並び替え用バッファ部13に供給し、保持させる。

【0147】

10

20

30

40

50

係数並び替え処理実行部 2 1 2 は、制御部 2 0 1 に制御されて、係数並び替え用バッファ部 1 3 に保持されている出力係数を読み出してエントロピ符号化部 1 5 に供給する、図 1 の係数並び替え部 1 4 の機能を実現する係数並び替え処理を実行する。係数並び替え処理実行部 2 1 2 は、係数並び替え用バッファ部 1 3 に保持されている出力係数を、符号化順（合成フィルタ処理順）に、出力係数読み出し部 2 1 7 を介して取得し、それをエントロピ符号化処理実行部 2 1 3 に供給することにより、係数の並び替えを行う。

【 0 1 4 8 】

エントロピ符号化処理実行部 2 1 3 は、制御部 2 0 1 に制御されて、係数並び替え部 1 4 より供給される係数データをその供給順に符号化する図 1 のエントロピ符号化部 1 5 の機能を実現するエントロピ符号化処理を実行する。エントロピ符号化処理実行部 2 1 3 は、係数並び替え処理実行部 2 1 2 より供給された係数データ（出力係数）を、その供給順に符号化し、得られた符号化データを、RAM 1 0 3、出力部 1 1 2、記憶部 1 1 3、通信部 1 1 4、またはリムーバブルメディア 1 2 1 等の CPU 1 0 1 の外部に出力する。

【 0 1 4 9 】

途中計算用データ読み出し部 2 1 4 および途中計算用データ書き込み部 2 1 5 は、制御部 2 0 1 に制御されて、キャッシュメモリ（図示せず）または RAM 1 0 3 等に対して、途中計算用データの読み出しや書き込みを行うことにより、図 1 の途中計算用バッファ部 1 2 を、キャッシュメモリ（図示せず）または RAM 1 0 3 等の記憶領域に形成させる。途中計算用データ読み出し部 2 1 4 は、途中計算用バッファ部 1 2 に保持されている途中計算用データ群の中から必要なものを読み出して分析フィルタ処理実行部 2 1 1 に供給する。途中計算用データ書き込み部 2 1 5 は、分析フィルタ処理実行部 2 1 1 より供給される途中計算用データを途中計算用バッファ部 1 2 に書き込む（保持させる）。

【 0 1 5 0 】

出力係数書き込み部 2 1 6 および出力係数読み出し部 2 1 7 は、制御部 2 0 1 に制御されて、キャッシュメモリ（図示せず）または RAM 1 0 3 等に対して、出力係数の読み出しや書き込みを行うことにより、図 1 の係数並び替え用バッファ部 1 3 を、キャッシュメモリ（図示せず）または RAM 1 0 3 等の記憶領域に形成させる。出力係数書き込み部 2 1 6 は、分析フィルタ処理実行部 2 1 1 より供給される出力係数を係数並び替え用バッファ部 1 3 に書き込む（保持させる）。出力係数読み出し部 2 1 7 は、係数並び替え用バッファ部 1 3 に保持されている出力係数を、係数並び替え処理実行部 2 1 2 に指定された順に読み出して係数並び替え処理実行部 2 1 2 に供給する。

【 0 1 5 1 】

画像データバッファ部 2 2 1 は、入力される画像データを一時的に保持するバッファ部であり、制御部 2 0 1 の制御により、例えば RAM 1 0 3 に形成される。画像データバッファ部 2 2 1 は、制御部 2 0 1 の制御されて、略定期的に 1 ラインずつ供給される画像データを一時的に保持し、所定のタイミングに合わせて、その保持している画像データを 1 ラインずつ分析フィルタ処理実行部 2 1 1 に供給する。

【 0 1 5 2 】

なお、図 1 4 においては、途中計算用バッファ部 1 2 と係数並び替え用バッファ部 1 3 を CPU 1 0 1 内部のキャッシュメモリ（図示せず）に形成し、画像データバッファ部 2 2 1 を RAM 1 0 3 に形成しているが、これは一例であり、途中計算用バッファ部 1 2、係数並び替え用バッファ部 1 3、および画像データバッファ部 2 2 1 は、いずれも、任意の記憶領域に形成することができる。ただし、上述したようにキャッシュメモリは、高速にアクセス可能であるものの、容量が RAM 1 0 3 に比べて小さいので、不要な負荷や遅延時間の増大を招く恐れのあるバッファ溢れ等を生じさせないようにするために、高頻度で読み書きされるデータや、近い将来に使用予定のあるデータのみを記憶させるようにするのが望ましい。例えば、アクセス頻度の多い途中計算用バッファ部 1 2 は、キャッシュメモリに形成し、アクセス頻度の少ない画像データバッファ部 2 2 1 や係数並び替え用バッファ部 1 3 は、RAM 1 0 3 に形成するようにしてもよい。

【 0 1 5 3 】

図 1 4 の制御部 2 0 1 は、図 1 の符号化部 1 0 を実現するために、各部を制御する制御処理を実行する。図 1 5 および図 1 6 のフローチャートを参照して、その制御処理の流れの例を説明する。

【 0 1 5 4 】

制御処理が開始されると、制御部 2 0 1 は、ステップ S 1 0 1 において、画像データバッファ部 2 2 1 を、例えば RAM 1 0 3 に形成させ、供給される画像データの蓄積を開始させる。このように制御された画像データバッファ部 2 2 1 は、これ以降、制御処理が終了されるまで、1 ラインずつ供給される画像データを一時的に保持する。

【 0 1 5 5 】

ステップ S 1 0 2 において、制御部 2 0 1 は、画像データバッファ部 2 2 1 に、所定ライン数分の画像データが蓄積されたか否かを判定し、蓄積されたと判定するまで待機する。分析フィルタ処理実行部 2 1 1 においてリフティング演算が可能な程度、画像データが蓄積されたと判定した場合、制御部 2 0 1 は、処理をステップ S 1 0 3 に進める。

【 0 1 5 6 】

ステップ S 1 0 3 において、制御部 2 0 1 は、分析フィルタ処理実行部 2 1 1 を制御し、画像データバッファ部 2 2 1 より、最下位レベルのリフティング演算 1 回、つまり、分割レベル 1 の分析フィルタ処理のリフティング演算 1 回に必要なライン数（例えば、2 ラインまたは 3 ライン）の画像データを取得させる。この制御により、分析フィルタ処理実行部 2 1 1 は、画像データバッファ部 2 2 1 より、最下位レベルのリフティング演算 1 回に必要なライン数の画像データを、ライン番号順に取得する。

【 0 1 5 7 】

ステップ S 1 0 4 において、制御部 2 0 1 は、分析フィルタ処理実行部 2 1 1 を制御し、途中計算用バッファ部 1 2 より、これから行うリフティング演算に必要な途中計算用データを取得させる。この制御により、分析フィルタ処理実行部 2 1 1 は、途中計算用データ読み出し部 2 1 4 を制御して、これから行うリフティング演算に必要な途中計算用データを、途中計算用バッファ部 1 2 より読み出させ、供給させる（つまり、途中計算用データを取得する）。この制御により、途中計算用データ読み出し部 2 1 4 は、要求された途中計算用データを途中計算用バッファ部 1 2 より読み出し、分析フィルタ処理実行部 2 1 1 に供給する。

【 0 1 5 8 】

ステップ S 1 0 5 において、制御部 2 0 1 は、分析フィルタ処理実行部 2 1 1 を制御し、リフティング演算 1 回分の分析フィルタ処理を実行させる。この制御により、分析フィルタ処理実行部 2 1 1 は、図 1 1 を参照して説明したように、実行可能な分析フィルタ処理のリフティング演算（曲線で囲まれた部分の演算）を 1 回分実行する。

【 0 1 5 9 】

リフティング演算が行われると、制御部 2 0 1 は、ステップ S 1 0 6 において、分析フィルタ処理実行部 2 1 1 を制御し、リフティング演算に用いられた画像データや係数データ、およびリフティング演算により得られた係数データのうち、次回以降のリフティング演算に利用するデータを途中計算用データとして途中計算用バッファ部 1 2 に保持させる。この制御により、分析フィルタ処理実行部 2 1 1 は、途中計算用データ書き込み部 2 1 5 に途中計算用データを供給するとともに、その途中計算用データ書き込み部 2 1 5 を制御し、供給した途中計算用データを途中計算用バッファ部 1 2 に書き込ませる。この制御により、途中計算用データ書き込み部 2 1 5 は、分析フィルタ処理実行部 2 1 1 より供給された途中計算用データを、途中計算用バッファ部 1 2 に書き込み、保持させる。

【 0 1 6 0 】

ステップ S 1 0 7 において、制御部 2 0 1 は、新たな画像データの入力無しに、さらにリフティング演算が可能であるか否かを判定する。可能であると判定した場合、制御部 2 0 1 は、処理をステップ S 1 0 8 に進め、分析フィルタ処理実行部 2 1 1 を制御し、リフティング演算により得られた出力係数を、係数並び替え用バッファ部 1 3 に保持させる。この制御により、分析フィルタ処理実行部 2 1 1 は、出力係数書き込み部 2 1 6 に出力係

10

20

30

40

50

数を供給するとともに、その出力係数書き込み部 2 1 6 を制御し、供給した出力係数を係数並び替え用バッファ部 1 3 に書き込ませる。この制御により、出力係数書き込み部 2 1 6 は、分析フィルタ処理実行部 2 1 1 より供給された出力係数を、係数並び替え用バッファ部 1 3 に書き込み、保持させる。

【 0 1 6 1 】

ステップ S 1 0 8 の処理が終了すると、制御部 2 0 1 は、処理をステップ S 1 0 4 に戻し、それ以降の処理を繰り返す。つまり、制御部 2 0 1 は、ステップ S 1 0 4 乃至ステップ S 1 0 8 の処理を繰り返すことにより、図 1 1 を参照して説明したように、入力された画像データに対応する、実行可能なリフティング演算を全て行わせる。

【 0 1 6 2 】

そして、ステップ S 1 0 7 において、実行可能なリフティング演算が無くなったと判定した場合、制御部 2 0 1 は、処理をステップ S 1 0 9 に進める。

【 0 1 6 3 】

ステップ S 1 0 9 において、制御部 2 0 1 は、1 プレシント分のフィルタ処理が終了したか否かを判定し、終了していないと判定した場合、処理をステップ S 1 1 0 に進め、ステップ S 1 0 8 の場合と同様に、分析フィルタ処理実行部 2 1 1 を制御し、リフティング演算により得られた出力係数を係数並び替え用バッファ部 1 3 に保持させる。ステップ S 1 1 0 の処理が終了すると、制御部 2 0 1 は、処理をステップ S 1 0 3 に戻し、それ以降の処理を繰り返す。つまり、制御部 2 0 1 は、ステップ S 1 0 3 乃至ステップ S 1 1 0 の処理を繰り返すことにより、図 1 1 を参照して説明したように、リフティング演算を 1 プレシント分実行させる。

【 0 1 6 4 】

そして、ステップ S 1 0 9 において、1 プレシント分のフィルタ処理が終了したと判定した場合、制御部 2 0 1 は、処理を図 1 6 のステップ S 1 2 1 に進める。図 1 6 のステップ S 1 2 1 において、制御部 2 0 1 は、エントロピ符号化処理実行部 2 1 3 を制御し、最後のリフティング演算の出力係数をエントロピ符号化させる。「最後のリフティング演算の出力係数」とは、上述した図 1 1 の説明において 5 番目に合成フィルタ処理される係数データおよび 6 番目に合成フィルタ処理される係数データに相当する。つまり、この最後のリフティング演算の出力係数は、係数並び替え用バッファ部 1 3 を介さずにエントロピ符号化させる。制御部 2 0 1 がこのように制御することにより、係数並び替え用バッファ部 1 3 として必要なメモリ量を低減させることができるとともに、負荷や遅延時間を低減させることができる。

【 0 1 6 5 】

ステップ S 1 2 1 の制御により、エントロピ符号化処理実行部 2 1 3 は、5 番目に合成フィルタ処理される係数データおよび 6 番目に合成フィルタ処理される係数データを順に符号化し、得られた符号化データを順次出力する。

【 0 1 6 6 】

その符号化処理が終了すると、制御部 2 0 1 は、ステップ S 1 2 2 において、係数並び替え処理実行部 2 1 2 を制御し、係数並び替え用バッファ部 1 3 の出力係数を所定順に読み出させる。この制御により、係数並び替え処理実行部 2 1 2 は、出力係数読み出し部 2 1 7 を制御し、係数並び替え用バッファ部 1 3 に保持されている係数データを、合成フィルタ処理を行う順に読み出させて供給させる。この制御により、出力係数読み出し部 2 1 7 は、係数並び替え用バッファ部 1 3 に保持されている係数データを、合成フィルタ処理を行う順に読み出し、読み出した係数データを係数並び替え処理実行部 2 1 2 に供給させる。係数並び替え処理実行部 2 1 2 は、供給された係数データを順次エントロピ符号化処理実行部 2 1 3 に供給する。

【 0 1 6 7 】

ステップ S 1 2 3 において、制御部 2 0 1 は、エントロピ符号化処理実行部 2 1 3 を制御し、係数並び替え用バッファ部 1 3 より読み出された係数データをエントロピ符号化させる。この制御により、エントロピ符号化処理実行部 2 1 3 は、読み出された係数データ

10

20

30

40

50

を符号化し、得られた符号化データを出力する。

【0168】

ステップS124において、制御部201は、係数並び替え用バッファ部13に、符号化処理していない出力係数が存在するか否かを判定し、存在すると判定した場合、処理をステップS122に戻し、それ以降の処理を繰り返す。つまり、制御部201は、ステップS122乃至ステップS124の処理を繰り返すことにより、係数並び替え用バッファ部13に保持されている全ての出力係数に対して符号化処理を実行させる。ステップS121の処理が開始された時点において、係数並び替え用バッファ部13には、1プレシント分の出力係数（最後のリフティング演算の出力係数は除く）が蓄積されている。制御部201は、この出力係数全てについて符号化処理を実行させる。

10

【0169】

ステップS124において、係数並び替え用バッファ部13に未処理の出力係数が存在しないと判定した場合、制御部201は、処理をステップS125に進める。

【0170】

ステップS125において、制御部201は、1ピクチャ分の分析フィルタ処理が終了したか否かを判定する。つまり、制御部201は、ピクチャ内の全てのプレシントについて分析フィルタ処理（およびエントロピ符号化処理）を行ったか否かを判定する。未処理のプレシントが存在すると判定した場合、制御部201は、処理を図15のステップS102に戻し、それ以降の処理を繰り返す。つまり、制御部201は、ステップS102乃至ステップS125の各処理を繰り返し実行することにより、全てのプレシントに対して上述したような制御処理を行う。

20

【0171】

そして、ステップS125において、1ピクチャ分の分析フィルタ処理が終了したと判定した場合、制御部201は、制御処理を終了する。なお、この制御処理は、画像データのピクチャ毎に繰り返し実行される。

【0172】

以上のように制御処理を行うことにより、制御部201は、分析フィルタ処理を効率よく行い、画像データバッファ部221として必要なメモリ容量を低減させる。また、制御部201は、最後のリフティング演算の出力係数を係数並び替え用バッファ部13に蓄積させないようにすることにより、係数並び替え用バッファ部13として必要なメモリ量を低減させるとともに、負荷や遅延時間を低減させることができる。

30

【0173】

なお、図15および図16においては、制御処理の流れの例を説明したが、分析フィルタ処理実行部211乃至出力係数読み出し部217は、上述したように、この制御処理の各ステップに対応して制御部201の制御に対応する処理を実行する。従って、図15および図16のフローチャートに示される制御処理の各ステップは、上述したとおりに、分析フィルタ処理実行部211乃至出力係数読み出し部217のそれぞれが実行する処理も表しているものとし、各部が実行する処理の図示は省略する。

【0174】

次に、他の処理進行方法について説明する。

40

【0175】

図11を参照して説明したように、復号部20における第2プレシントに対する符号化処理は、符号化部10における第2プレシントに対するウェーブレット変換処理が終了するまで開始することができない。そのため、図12を参照して説明したように、CPU101は、ウェーブレット変換処理とエントロピ符号化処理をプレシント単位で交互に実行することになる。しかしながら、例えば、CPU101が各分析フィルタ処理をより高速に処理し、画像データ入力タイミング（水平同期タイミング）の間隔よりも早く終了させることができる場合、次の水平同期タイミング（分析フィルタ処理開始タイミング）まで待機時間が生じてしまい、CPU101の使用効率が低下してしまう。

【0176】

50

例えば、図 1 2 においても、ライン番号 2 1（およびライン番号 2 0）の画像データが入力されてから、分析フィルタ処理は 3 回行われるのに対し、ライン番号 1 6（およびライン番号 1 5）の画像データが入力されたときは、分析フィルタ処理は 1 回しか行われない。つまり、次のライン番号 1 8（およびライン番号 1 7）の画像データが入力されるまでに待機時間が生じる。このような待機時間が生じると、CPU 1 0 1 の使用効率が低下することになり、その効率が低い分だけ、処理実行時の CPU 1 0 1 にかかる負荷が増大することになる。従って、その分高性能な CPU を用いなければならず、製造コストや消費電力の増大にも繋がる。つまり、CPU 1 0 1 にかかる負荷を低減させるためには、CPU 1 0 1 の使用効率をより向上させる（より効率よく CPU 1 0 1 を動作させる）ことが求められる。

【 0 1 7 7 】

10

そこで、このような待機時間の発生を抑制し、CPU 1 0 1 の使用効率を向上させるために、図 1 7 に示されるように、エントロピ符号化処理を行うプレシントを、ウェーブレット変換処理を行うプレシントよりも 1 つ遅延させる。換言すれば、符号化処理が、常に、リフティング演算により最後に生成された最低域の周波数成分よりも先に生成された周波数成分を符号化するようにする。さらに換言すれば、符号化処理の進行に対してウェーブレット変換処理の進行を、図 1 1 を参照して説明した例の場合よりも早めるようにし、符号化処理が最低域の周波数成分を符号化する前に、ウェーブレット変換処理がその最低域の周波数成分を生成しているようにし、符号化処理が常に次の周波数成分を符号化することができるようにする。

【 0 1 7 8 】

20

図 1 7 の例においては、符号化部 1 0 における第 3 プレシントに対してウェーブレット変換処理を行いながら、復号部 2 0 における第 2 プレシントに対して符号化処理を行う様子を示している。もちろん、他のプレシントについても同様であるのでその説明は省略する。

【 0 1 7 9 】

なお、図 1 7 は、図 1 1 に対応する図であるので、その詳細についての説明は省略する。図 1 7 において、曲線 2 5 1 は、符号化部 1 0 における第 1 プレシント（ライン番号 0 乃至ライン番号 1 4）と第 2 プレシント（ライン番号 1 5 乃至ライン番号 2 2）との境界を示し、曲線 2 5 2 は、符号化部 1 0 における第 2 プレシント（ライン番号 1 5 乃至ライン番号 2 2）と第 3 プレシント（ライン番号 2 3 乃至ライン番号 3 0）との境界を示している。

30

【 0 1 8 0 】

図 1 7 に示されるように、符号化部 1 0 における第 3 プレシントに対するウェーブレット変換処理により生成される出力係数（曲線 2 6 1 において囲まれる部分の黒丸（ ）や黒四角（ ））と、復号部 2 0 における第 2 プレシントに対するエントロピ符号化において処理される出力係数（曲線 1 5 2 において囲まれる部分の黒丸（ ）や黒四角（ ））とは、互いに重複しない。従って、符号化部 1 0 における第 3 プレシントに対するウェーブレット変換処理と、復号部 2 0 における第 2 プレシントに対するエントロピ符号化を同時並行的に行うことができる。なお、CPU 1 0 1 は 1 つの演算処理部であるため、ここでいう「同時並行的」とは、2 つの処理を実際に同時に並行して行うことではなく、2 つの処理の開始から終了までの時間が互いに重複することを示す。より具体的に説明すると、CPU 1 0 1 を時分割して、ウェーブレット変換処理とエントロピ符号化処理のそれぞれを、より細かい処理単位で交互に実行させる。このようにすることにより、ウェーブレット変換処理の開始から終了までの時間と、エントロピ符号化処理の開始から終了までの時間が互いに重複するので、CPU 1 0 1 は、見かけ上、2 つの処理を「同時並行的に」実行することができる。

40

【 0 1 8 1 】

つまり、図 1 7 の例においては、CPU 1 0 1 は、符号化部 1 0 における第 3 プレシントに対するウェーブレット変換処理として、ライン番号 2 3 およびライン番号 2 4 の画像データが入力されると、曲線 2 7 1 により囲まれるリフティング演算を行い、ライン番号

50

25 およびライン番号26の画像データが入力されると、曲線272により囲まれるリフティング演算を行い、ライン番号27およびライン番号28の画像データが入力されると、曲線273により囲まれるリフティング演算を行い、ライン番号29およびライン番号30の画像データが入力されると、曲線274により囲まれるリフティング演算を行う。これらの各処理の間に発生するCPU101の待機時間中に、曲線152により囲まれる部分の、復号部20における第2プレシнктに対するエントロピ符号化処理を、処理可能なライン数分だけCPU101に実行させるようにする。

【0182】

図18に、この場合の、CPU101において実行される処理の流れを模式的に示す。図18は、図12に対応する図であり、基本的には図12と同様の構成であるので、その詳細についての説明は省略する。

10

【0183】

図18においても、CPU101は、図12の場合と同様に、符号化部10における第3プレシнктについて、画像データが2ライン入力される毎に実行可能なリフティング演算（分析フィルタ処理）を行う（DWT（Lev1）乃至DWT（Lev3））。ただし、図18の場合、CPU101は、リフティング演算終了後、次の2ラインが入力されるまでの間に（次のリフティング演算実行開始タイミングまでの間に）、復号部20における第2プレシнктに対するエントロピ符号化処理を、係数データのライン単位で実行する。

【0184】

つまり、図18において、ライン番号24（および23）の画像データが入力されると、CPU101は、図17の曲線271に囲まれた部分のリフティング演算DWT（Lev1）を行い、27番目に合成フィルタ処理される係数データを生成し、それを係数並び替え用バッファ部13に保持させる。その後、CPU101は、次のリフティング演算実行開始タイミングまでの時間を利用して、5番目および6番目に合成フィルタ処理される係数データを順次エントロピ符号化し（VLC5およびVLC6）、それぞれの処理において得られた符号化データを順次出力する。

20

【0185】

そして、ライン番号26（および25）の画像データが入力されると、CPU101は、図17の曲線272に囲まれた部分のリフティング演算DWT（Lev1）およびDWT（Lev2）を行い、それぞれの処理において28番目に合成フィルタ処理される係数データと23番目に合成フィルタ処理される係数データを生成し、それらを順次係数並び替え用バッファ部13に保持させる。その後、CPU101は、次のリフティング演算実行開始タイミングまでの時間を利用して、7番目乃至9番目に合成フィルタ処理される係数データを順次エントロピ符号化し（VLC7、VLC8、およびVLC9）、それぞれの処理において得られた符号化データを順次出力する。

30

【0186】

次に、ライン番号28（および27）の画像データが入力されると、CPU101は、図17の曲線273に囲まれた部分のリフティング演算DWT（Lev1）を行い、31番目に合成フィルタ処理される係数データを生成し、それを係数並び替え用バッファ部13に保持させる。その後、CPU101は、次のリフティング演算実行開始タイミングまでの時間を利用して、10番目乃至12番目に合成フィルタ処理される係数データを順次エントロピ符号化し（VLC10、VLC11、およびVLC12）、それぞれの処理において得られた符号化データを順次出力する。

40

【0187】

そして、ライン番号30（および29）の画像データが入力されると、CPU101は、図17の曲線274に囲まれた部分のリフティング演算DWT（Lev1）、DWT（Lev2）、およびDWT（Lev3）を行い、それぞれの処理において25番目に合成フィルタ処理される係数データ、18番目に合成フィルタ処理される係数データ、21番目に合成フィルタ処理される係数データ、および22番目に合成フィルタ処理される係数データを生成し、それらを順次係数並び替え用バッファ部13に保持させる。

50

【 0 1 8 8 】

このように、エントロピ符号化処理の処理対象プレシントを、ウェーブレット変換処理の処理対象プレシントに対して1つ遅らせる、すなわち、符号化処理が、常に、リフティング演算により最後に生成された最低域の周波数成分より先に生成された周波数成分を符号化するようにすることにより、CPU 1 0 1 は、ウェーブレット変換処理の空き時間を利用して、エントロピ符号化処理を実行することができ、見かけ上、ウェーブレット変換処理とエントロピ符号化処理を同時並行的に実行することができる。このようにすることにより、CPU 1 0 1 の待機時間を低減させることができ、CPU 1 0 1 の使用効率を向上させることができる。

【 0 1 8 9 】

また、これにより、CPU 1 0 1 は、容易に分析フィルタ処理をリアルタイム（即時的）に行うことができる。図 1 9 に示されるように、CPU 1 0 1 は、2 ライン分の画像データの入力タイミング（hsync）に合わせて、1 プレシントに対するウェーブレット変換処理を複数回に分けて実行する（P1 DWT、P2 DWT、P3 DWT、・・・、Pn DWT）。CPU 1 0 1 は、これらのウェーブレット変換処理の空き時間に、各出力係数に対する符号化処理を全て行う（P1 VLC、P2 VLC、・・・、Pn - 1 VLC、Pn VLC）。このようにエントロピ符号化処理を、各出力係数に対する処理を処理単位として、ウェーブレット変換処理の空き時間に行うことにより、高性能なCPU 1 0 1 でなくても、容易にウェーブレット変換をリアルタイム（即時的）に行うことができる。

【 0 1 9 0 】

従って、この場合、入力される画像データをバッファする必要が無く、画像データバッファ部 2 2 1 が不要になる。従って、図 1 2 乃至図 1 6 を参照して説明した方法の場合よりも、符号化部 1 0 が行う処理に必要なメモリ量を低減させることができるだけでなく、画像データをバッファする時間を省略することができるので、ウェーブレット変換処理およびエントロピ符号化処理の負荷や遅延時間を低減させることができる。

【 0 1 9 1 】

なお、図 1 2 の場合と図 1 8 の場合を比較すると、図 1 8 の場合の方が、ライン番号 2 4（および 2 3）の画像データが入力された後に行われるリフティング演算DWT（Lev 1）の分だけ、5 番目に合成フィルタ処理される係数データを符号化して得られる符号化データの出力タイミングが遅く、また、5 番目、6 番目、および 2 7 番目に合成フィルタ処理される係数データの分だけ、係数並び替え用バッファ部 1 3 に蓄積されるデータ量が多い。しかしながら、このリフティング演算DWT（Lev 1）は、分割レベル 1 のリフティング演算 1 回分の遅延時間であり、図 1 8 に示されるように 2 ライン分の画像データの入力タイミング（hsync）の間隔よりは短い。また、係数並び替え用バッファ部 1 3 に蓄積される 5 番目に合成フィルタ処理される係数データは分割レベル 3 の低域成分 1 ライン分であり、6 番目に合成フィルタ処理される係数データは分割レベル 3 の高域成分 1 ライン分であり、2 7 番目に合成フィルタ処理される係数データは分割レベル 1 の高域成分 1 ライン分であるので、そのデータ量は画像データ 1 ライン分より少ない（図 4 参照）。

【 0 1 9 2 】

図 1 2 の方法において、エントロピ符号化によりウェーブレット変換処理をリアルタイムに行うことができない場合、1 プレシント分のエントロピ符号化処理の処理時間は、2 ライン分の画像データの入力タイミング（hsync）の間隔より長い。また、そのときの画像データバッファ部 2 2 1 に蓄積される画像データのデータ量は、画像データ 2 ライン分以上である。

【 0 1 9 3 】

従って、この入力画像データのバッファ分まで含めて比較すると、図 1 8 の場合の方が、図 1 2 の場合よりも、必要とするメモリ容量が少なく、かつ、遅延時間が短い。また、図 1 2 の場合よりもCPU 1 0 1 の使用効率が高いため、より小さな負荷で、所定の速度で入力される画像データをリアルタイムに処理することができる。

【 0 1 9 4 】

なお、図 18 の例の場合、5 番目と 6 番目に合成フィルタ処理される係数データを係数並び替え用バッファ部 13 に蓄積するように説明したが、図 12 の場合と同様に、これらの係数データを係数並び替え用バッファ部 13 に蓄積させずにエントロピ符号化処理するようにしてもよい。つまり、この場合、CPU 101 は、符号化部 10 における第 2 プレシントの最後のリフティング演算終了後、第 3 プレシントに対するウェーブレット変換を開始する前に、復号部 20 における第 2 プレシントに対するエントロピ符号化処理を開始することができる。このようにすることにより、CPU 101 は、5 番目に合成フィルタ処理される係数データを符号化して得られる符号化データの出力タイミングをさらに早くすることができ、また、係数並び替え用バッファ部 13 として必要なメモリ容量をさらに低減させることができる。

10

【0195】

また、図 18 の例の場合、図 19 に示されるように符号化データの出力タイミングが分散される。従って、後段の処理の負荷を均一化させることができるので、再生ビデオ信号のように略一定の速度で入力される画像データ（1 ラインずつ所定の間隔で入力される画像データ）に対してリアルタイムに処理を行うことが容易になる。例えば、符号化部 10 の後段の処理として符号化データをパケット化する場合、図 18 の例のように符号化データの生成レートが略一定であると、そのパケット化処理の負荷の変化量が低減されるので、パケット処理が容易になる。これに対して、図 12 の例の場合、符号化データの出力タイミングは、エントロピ符号化処理が行われている期間に集中し、ムラが生じる。従って、パケット化処理の負荷の変化量が増大し、負荷がピークのときにオーバーフローしないようにパケット化処理を高速に実行するようにしたり、符号化データをバッファしたりする必要が生じ、それによりコスト、負荷、および遅延時間の増大を招く恐れがある。

20

【0196】

なお、図 18 の例は一例であり、エントロピ符号化処理は、リフティング処理の空き時間であれば任意に割り当てることができる。例えば、図 18 においては、ライン番号 24（および 23）の画像データが入力された後に実行されるリフティング演算 DWT（Lev 1）の後の空き時間に、5 番目と 6 番目に合成フィルタ処理される係数データの符号化（VLC 5 および VLC 6）を割り当てるように説明したが、さらに 7 番目以降に合成フィルタ処理される係数データの符号化（VLC 7 乃至 VLC 12）を割り当てるようにしてもよい。また、ライン番号 30（および 29）の画像データが入力された後に実行されるリフティング演算 DWT（Lev 1）、DWT（Lev 2）、および DWT（Lev 3）の後の空き時間に符号化を割り当てるようにしてもよい。

30

【0197】

次に、このような流れで処理を実行する符号化部 10 の実現化について、より具体的に説明する。図 20 は、上述した流れで処理を実行する符号化部 10 を実現するソフトウェアプログラムを実行する CPU 101 が有する機能を示す機能ブロック図である。

【0198】

図 20 は、図 14 に対応する図であり、図 14 に示されるブロックと対応するブロックには同一の番号を付してある。図 20 の場合、図 14 の場合より画像データバッファ部 221 が省略されるのみであり、CPU 101 が有する機能ブロックは基本的に図 14 の場合と同様であり、制御部 201、分析フィルタ処理実行部 211、係数並び替え処理実行部 212、エントロピ符号化処理実行部 213、途中計算用データ読み出し部 214、途中計算用データ書き込み部 215、出力係数書き込み部 216、および出力係数読み出し部 217 を有する。ただし、図 20 の場合、制御部 201 は、画像データバッファ部 221 に関する制御は行わない。

40

【0199】

図 20 の制御部 201 は、図 1 の符号化部 10 を実現するために、各部を制御する制御処理を実行する。図 21 のフローチャートを参照して、その制御処理の流れの例を説明する。なお、この制御処理は、画像データのピクチャ毎に繰り返し実行される。

【0200】

50

制御処理が開始されると、制御部 201 は、ステップ S 201 において、所定ライン数（例えば 2 ラインまたは 3 ライン）分の画像データが入力されたか否かを判定し、入力されたと判定するまで待機する。そして、入力されたと判定した場合、制御部 201 は、処理をステップ S 202 に進める。

【0201】

ステップ S 202 において、制御部 201 は、分析フィルタ処理実行部 211 を制御し、途中計算用バッファ部 12 より、これから行うリフティング演算に必要な途中計算用データを取得させる。この制御により、分析フィルタ処理実行部 211 は、途中計算用データ読み出し部 214 を制御して、これから行うリフティング演算に必要な途中計算用データを、途中計算用バッファ部 12 より読み出させ、供給させる（つまり、途中計算用データを取得する）。この制御により、途中計算用データ読み出し部 214 は、要求された途中計算用データを途中計算用バッファ部 12 より読み出し、分析フィルタ処理実行部 211 に供給する。

10

【0202】

ステップ S 203 において、制御部 201 は、分析フィルタ処理実行部 211 を制御し、リフティング演算 1 回分の分析フィルタ処理を実行させる。この制御により、分析フィルタ処理実行部 211 は、図 18 を参照して説明したように、実行可能な分析フィルタ処理のリフティング演算を 1 回分実行する。

【0203】

リフティング演算が行われると、制御部 201 は、ステップ S 204 において、分析フィルタ処理実行部 211 を制御し、リフティング演算に用いられた画像データや係数データ、およびリフティング演算により得られた係数データのうち、次回以降のリフティング演算に利用するデータを途中計算用データとして途中計算用バッファ部 12 に保持させる。この制御により、分析フィルタ処理実行部 211 は、途中計算用データ書き込み部 215 に途中計算用データを供給するとともに、その途中計算用データ書き込み部 215 を制御し、供給した途中計算用データを途中計算用バッファ部 12 に書き込ませる。この制御により、途中計算用データ書き込み部 215 は、分析フィルタ処理実行部 211 より供給された途中計算用データを、途中計算用バッファ部 12 に書き込み、保持させる。

20

【0204】

ステップ S 205 において、制御部 201 は、分析フィルタ処理実行部 211 を制御し、リフティング演算により得られた出力係数を、係数並び替え用バッファ部 13 に保持させる。この制御により、分析フィルタ処理実行部 211 は、出力係数書き込み部 216 に出力係数を供給するとともに、その出力係数書き込み部 216 を制御し、供給した出力係数を係数並び替え用バッファ部 13 に書き込ませる。この制御により、出力係数書き込み部 216 は、分析フィルタ処理実行部 211 より供給された出力係数を、係数並び替え用バッファ部 13 に書き込み、保持させる。

30

【0205】

ステップ S 206 において、制御部 201 は、新たな画像データの入力無しに、さらにリフティング演算が可能であるか否かを判定し、可能であると判定した場合、処理をステップ S 202 に戻し、それ以降の処理を繰り返す。つまり、制御部 201 は、ステップ S 202 乃至ステップ S 206 の処理を繰り返すことにより、図 18 を参照して説明したように、入力された画像データに対応する、実行可能なリフティング演算を全て行わせる。

40

【0206】

そして、ステップ S 206 において、実行可能なリフティング演算が無くなったと判定した場合、制御部 201 は、処理をステップ S 207 に進める。

【0207】

ステップ S 207 において、制御部 201 は、1 ピクチャ分のフィルタ処理が終了したか否かを判定し、終了していないと判定した場合、処理をステップ S 208 に進め、係数並び替え用バッファ部 13 に保持されている、1 つ前のプレシントより処理対象出力係数群を特定する。つまり、制御部 201 は、次のリフティング演算開始タイミングまでの

50

空き時間に符号化する出力係数を、リフティング演算により最後に生成された最低域の周波数成分より先に生成された周波数成分の中から特定する。なお、分析フィルタ処理されたのが第1プレシンクトである場合、制御部201は、1つ前のピクチャの最後のプレシンクトを1つ前のプレシンクトとし、処理対象出力係数群を特定する。また、分析フィルタ処理されたのが先頭のピクチャの第1プレシンクトである場合、制御部201は、ステップS208乃至ステップS211の処理を省略し、処理をステップS201に戻し、それ以降の処理を実行させる。

【0208】

ステップS209において、制御部201は、係数並び替え処理実行部212を制御し、処理対象出力係数として指定された係数データを所定順に読み出させる。この制御により、係数並び替え処理実行部212は、出力係数読み出し部217を制御し、係数並び替え用バッファ部13に保持されている係数データ（処理対象出力係数）を、合成フィルタ処理を行う順に読み出させて供給させる。この制御により、出力係数読み出し部217は、係数並び替え用バッファ部13に保持されている係数データを、合成フィルタ処理を行う順に読み出し、読み出した係数データを係数並び替え処理実行部212に供給させる。係数並び替え処理実行部212は、供給された係数データを順次エントロピ符号化処理実行部213に供給する。

10

【0209】

ステップS210において、制御部201は、エントロピ符号化処理実行部213を制御し、係数並び替え用バッファ部13より読み出された係数データをエントロピ符号化させる。この制御により、エントロピ符号化処理実行部213は、読み出された係数データを符号化し、得られた符号化データを出力する。

20

【0210】

ステップS211において、制御部201は、係数並び替え用バッファ部13に、符号化処理していない処理対象出力係数が存在するか否かを判定し、存在すると判定した場合、処理をステップS209に戻し、それ以降の処理を繰り返す。つまり、制御部201は、ステップS209乃至ステップS211の処理を繰り返すことにより、係数並び替え用バッファ部13に保持されている全ての処理対象出力係数に対して符号化処理を実行させる。

【0211】

ステップS211において、係数並び替え用バッファ部13に未処理の処理対象出力係数が存在しないと判定した場合、制御部201は、処理をステップS201に戻し、それ以降の処理を繰り返す。つまり、制御部201は、ステップS201乃至ステップS211の処理を繰り返すことにより、ピクチャ内の全プレシンクトについて、ウェーブレット変換処理を実行させ、さらにその空き時間にエントロピ符号化処理を実行させることにより、ウェーブレット変換処理とエントロピ符号化処理を見かけ上、同時並行的に実行させる。

30

【0212】

ステップS207において、1ピクチャ分のフィルタ処理が終了したと判定した場合、制御部201は、制御処理を終了する。なお、この制御処理が、画像データの最後のピクチャに対するものである場合、制御部201は、係数並び替え用バッファ部13に保持されている残りの出力係数全てを処理対象出力係数群として特定し、ステップS209およびステップS211と同様の処理を行い、全出力係数を符号化し、その符号化データを出力してから制御処理を終了する。

40

【0213】

以上のように制御処理を行うことにより、制御部201は、容易に、ウェーブレット変換処理とエントロピ符号化処理を同時並行的に実行し、所定の速度で入力される画像データよりリアルタイムに符号化データを生成して出力することができる。また、これにより、入力画像データの保持が不要になる。このように、ウェーブレット変換処理および符号化処理の負荷を低減させることができる。

50

【 0 2 1 4 】

なお、図 2 1 においては、制御処理の流れの例を説明したが、分析フィルタ処理実行部 2 1 1 乃至出力係数読み出し部 2 1 7 は、上述したように、この制御処理の各ステップに対応して制御部 2 0 1 の制御に対応する処理を実行する。従って、図 2 1 のフローチャートに示される制御処理の各ステップは、上述したとおりに、分析フィルタ処理実行部 2 1 1 乃至出力係数読み出し部 2 1 7 のそれぞれが実行する処理も表しているものとし、各部が実行する処理の図示は省略する。

【 0 2 1 5 】

次に、他の処理進行方法について説明する。

【 0 2 1 6 】

図 1 1 を参照して説明したように、復号部 2 0 における第 2 プレシントに対する符号化処理は、符号化部 1 0 における第 2 プレシントに対するウェーブレット変換処理が終了するまで開始することができない。そこで、図 2 2 に示されるように、符号化データを保持するコードバッファ部 3 1 1 を設け、符号化部 1 0 における第 2 プレシントに対するウェーブレット変換処理の最後のリフティング演算よりも前に、それ以前に生成される係数データを符号化するようにしてもよい。つまり、この場合、CPU 1 0 1 は、合成フィルタ処理の処理順と異なる順序でエントロピ符号化処理を先に行い、得られた符号化データを保持することにより、符号化データを合成フィルタ処理の処理順に出力させる。

【 0 2 1 7 】

図 2 2 に、この場合の、CPU 1 0 1 において実行される処理の流れを模式的に示す。図 2 2 は、図 1 2 に対応する図であり、基本的には図 1 2 と同様の構成であるので、その詳細についての説明は省略する。

【 0 2 1 8 】

図 2 2 においても、CPU 1 0 1 は、図 1 2 の場合と同様に、符号化部 1 0 における第 2 プレシントについて、画像データが 2 ライン入力される毎に実行可能なリフティング演算（分析フィルタ処理）を行う（DWT（Lev 1）乃至 DWT（Lev 3））。ただし、図 2 2 の場合、CPU 1 0 1 は、図 1 8 の場合と同様に、リフティング演算終了後、次の 2 ラインが入力されるまでの間に（次のリフティング演算実行開始タイミングまでの間に）、復号部 2 0 における第 2 プレシントに対するエントロピ符号化処理を、係数データのライン単位で実行する。

【 0 2 1 9 】

ただし、図 2 2 の場合、このエントロピ符号化処理を、ウェーブレット変換の処理対象プレシントと同じプレシントに対して行う。すなわち、この場合のウェーブレット変換とエントロピ符号化処理の関係は、図 1 7 ではなく図 1 1 に示される例のようになる。つまり、上述したように、第 2 プレシントに対する分析フィルタ処理が開始された時点では、5 番目および 6 番目に合成フィルタ処理される係数データは生成されていない。図 2 2 の場合、CPU 1 0 1 は、これらの係数データの符号化よりも先に、既に生成され、係数並び替え用バッファ部 1 3 に保持されている、復号部 2 0 における第 2 プレシントの他の係数データ（すなわち、5 番目および 6 番目に合成フィルタ処理される係数データに対応し、それらよりも高域成分である、7 番目乃至 12 番目に合成フィルタ処理される係数データ）の符号化を行う。

【 0 2 2 0 】

図 2 2 の場合、これまでの例と異なり、符号化データを保持するコードバッファ部 3 1 1 が設けられている。処理対象プレシントの最後のリフティング演算により生成される係数データよりも先に符号化された、処理対象プレシントの他の係数データは、このコードバッファ部 3 1 1 に一旦保持される。そして、処理対象プレシントの最後のリフティング演算により生成される係数データが符号化されて出力されると、コードバッファ部 3 1 1 に保持されている符号化データが、合成フィルタ処理が行われる順に読み出されて出力される。

【 0 2 2 1 】

より具体的に説明すると、図 2 2 の例の場合、ライン番号 1 6 (および 1 5) の画像データが入力されると、CPU 1 0 1 は、図 1 1 の曲線 1 6 1 に囲まれた部分のリフティング演算 DWT (Lev 1) を行い、1 9 番目に合成フィルタ処理される係数データを生成し、それを係数並び替え用バッファ部 1 3 に保持させる。この後、CPU 1 0 1 は、次のリフティング演算実行開始タイミングまでの時間を利用して、7 番目乃至 9 番目に合成フィルタ処理される係数データを順次エントロピ符号化し (VLC 7、VLC 8、および VLC 9)、それぞれの処理において得られた符号化データをコードバッファ部 3 1 1 に保持させる。

【 0 2 2 2 】

そして、ライン番号 1 8 (および 1 7) の画像データが入力されると、CPU 1 0 1 は、図 1 1 の曲線 1 6 2 に囲まれた部分のリフティング演算 DWT (Lev 1) および DWT (Lev 2) を行い、2 0 番目および 1 5 番目に合成フィルタ処理される係数データを生成し、それらを順次係数並び替え用バッファ部 1 3 に保持させる。

【 0 2 2 3 】

さらに、ライン番号 2 0 (および 1 9) の画像データが入力されると、CPU 1 0 1 は、図 1 1 の曲線 1 6 3 に囲まれた部分のリフティング演算 DWT (Lev 1) を行い、2 4 番目に合成フィルタ処理される係数データを生成し、それを係数並び替え用バッファ部 1 3 に保持させる。この後、CPU 1 0 1 は、次のリフティング演算実行開始タイミングまでの時間を利用して、1 0 番目乃至 1 2 番目に合成フィルタ処理される係数データを順次エントロピ符号化し (VLC 1 0、VLC 1 1、および VLC 1 2)、それぞれの処理において得られた符号化データをコードバッファ部 3 1 1 に保持させる。

【 0 2 2 4 】

また、ライン番号 2 2 (および 2 1) の画像データが入力されると、CPU 1 0 1 は、図 1 1 の曲線 1 6 4 に囲まれた部分のリフティング演算 DWT (Lev 1)、DWT (Lev 2)、および DWT (Lev 3) を行い、2 5 番目、1 8 番目、5 番目、および 6 番目に合成フィルタ処理される係数データを生成し、2 5 番目および 1 8 番目に合成フィルタ処理される係数データを順次係数並び替え用バッファ部 1 3 に保持させる。この後、CPU 1 0 1 は、次のリフティング演算実行開始タイミングまでの時間を利用して、5 番目および 6 番目に合成フィルタ処理される係数データを順次エントロピ符号化し (VLC 5 および VLC 6)、それぞれの処理において得られた符号化データを出力する。そして、CPU 1 0 1 は、コードバッファ部 3 1 1 に保持させた 7 番目乃至 1 2 番目に合成フィルタ処理される係数データの符号化データを合成フィルタ処理順に読み出し、5 番目および 6 番目に合成フィルタ処理される係数データの符号化データに続けて出力する。

【 0 2 2 5 】

つまり、図 2 3 に示されるように、所定の速度で入力される画像データに対して、ウェーブレット変換処理とエントロピ符号化処理を、プレシントをずらすことなく、リアルタイムに実行することができる。このようにすることにより、CPU 1 0 1 は、生成した符号化データを、次のプレシントに対する分析フィルタ処理が開始される前に出力することができ、各符号化データの出力タイミングを、図 1 2 の場合と同じまま CPU 1 0 1 の使用効率を向上させることができる。さらに、図 2 2 の例の場合、画像データバッファ部 2 2 1 を省略することができる。

【 0 2 2 6 】

ただし、図 2 2 の例の場合、コードバッファ部 3 1 1 が必要になるので、その分、必要なメモリ容量が図 1 8 の例の場合よりも増大する。ただし、コードバッファ部 3 1 1 においては、符号化データを蓄積するため、画像データを蓄積する場合よりもそのデータ量が低減される。すなわち、図 1 2 の場合よりも、データのバッファに必要なメモリ容量を低減させることができる。

【 0 2 2 7 】

図 1 8 の場合と比較すると、最初のピクチャの先頭のプレシントよりエントロピ符号化処理を行うことができるので、CPU 1 0 1 の使用効率は図 2 2 の場合の方が高い。ただし、コードバッファ部 3 1 1 の分、必要なメモリ容量が増大する。また、符号化データの

10

20

30

40

50

出力タイミングにムラがあるので、符号化部 10 の後段の処理が図 22 の場合の方が困難になる（負荷が大きくなる恐れがある）。

【0228】

次に、このような流れで処理を実行する符号化部 10 の実現化について、より具体的に説明する。図 24 は、上述した流れで処理を実行する符号化部 10 を実現するソフトウェアプログラムを実行する CPU 101 が有する機能を示す機能ブロック図である。

【0229】

図 24 も、図 14 に対応する図であり、図 14 に示されるブロックと対応するブロックには同一の番号を付してある。図 24 の場合、CPU 101 は、図 20 の場合と基本的に同様の機能ブロックを有するが、さらに、コードバッファ部 311、符号化データ書き込み部 312、および符号化データ読み出し部 313 を有する。

【0230】

符号化データ書き込み部 312 および符号化データ読み出し部 313 は、制御部 201 に制御されて、キャッシュメモリ（図示せず）または RAM 103 等に対して、エン트로ピ符号化処理において生成された符号化データの読み出しや書き込みを行うことにより、図 22 のコードバッファ部 311 を、キャッシュメモリ（図示せず）または RAM 103 等の記憶領域に形成させる。符号化データ書き込み部 312 は、エン트로ピ符号化処理実行部 213 より供給される符号化データをコードバッファ部 311 に書き込む（保持させる）。よび符号化データ読み出し部 313 は、コードバッファ部 311 に保持されている符号化データを、合成フィルタ処理の順に読み出してエン트로ピ符号化処理実行部 213 に供給する。

【0231】

図 24 の制御部 201 は、図 1 の符号化部 10 を実現するために、各部を制御する制御処理を実行する。図 25 および図 26 のフローチャートを参照して、その制御処理の流れの例を説明する。なお、この制御処理は、画像データのピクチャ毎に繰り返し実行される。

【0232】

制御処理が開始されると、制御部 201 は、ステップ S301 において、所定ライン数（例えば 2 ラインまたは 3 ライン）分の画像データが入力されたか否かを判定し、入力されたと判定するまで待機する。そして、入力されたと判定した場合、制御部 201 は、処理をステップ S302 に進める。

【0233】

ステップ S302 において、制御部 201 は、分析フィルタ処理実行部 211 を制御し、途中計算用バッファ部 12 より、これから行うリフティング演算に必要な途中計算用データを取得させる。この制御により、分析フィルタ処理実行部 211 は、途中計算用データ読み出し部 214 を制御して、これから行うリフティング演算に必要な途中計算用データを、途中計算用バッファ部 12 より読み出させ、供給させる（つまり、途中計算用データを取得する）。この制御により、途中計算用データ読み出し部 214 は、要求された途中計算用データを途中計算用バッファ部 12 より読み出し、分析フィルタ処理実行部 211 に供給する。

【0234】

ステップ S303 において、制御部 201 は、分析フィルタ処理実行部 211 を制御し、リフティング演算 1 回分の分析フィルタ処理を実行させる。この制御により、分析フィルタ処理実行部 211 は、図 22 を参照して説明したように、実行可能な分析フィルタ処理のリフティング演算を 1 回分実行する。

【0235】

リフティング演算が行われると、制御部 201 は、ステップ S304 において、分析フィルタ処理実行部 211 を制御し、リフティング演算に用いられた画像データや係数データ、およびリフティング演算により得られた係数データのうち、次回以降のリフティング演算に利用するデータを途中計算用データとして途中計算用バッファ部 12 に保持させる

。この制御により、分析フィルタ処理実行部 2 1 1 は、途中計算用データ書き込み部 2 1 5 に途中計算用データを供給するとともに、その途中計算用データ書き込み部 2 1 5 を制御し、供給した途中計算用データを途中計算用バッファ部 1 2 に書き込ませる。この制御により、途中計算用データ書き込み部 2 1 5 は、分析フィルタ処理実行部 2 1 1 より供給された途中計算用データを、途中計算用バッファ部 1 2 に書き込み、保持させる。

【 0 2 3 6 】

ステップ S 3 0 5 において、制御部 2 0 1 は、新たな画像データの入力無しに、さらにリフティング演算が可能であるか否かを判定し、可能であると判定した場合、処理をステップ S 3 0 6 に進める。

【 0 2 3 7 】

ステップ S 3 0 6 において、制御部 2 0 1 は、分析フィルタ処理実行部 2 1 1 を制御し、リフティング演算により得られた出力係数を、係数並び替え用バッファ部 1 3 に保持させる。この制御により、分析フィルタ処理実行部 2 1 1 は、出力係数書き込み部 2 1 6 に出力係数を供給するとともに、その出力係数書き込み部 2 1 6 を制御し、供給した出力係数を係数並び替え用バッファ部 1 3 に書き込ませる。この制御により、出力係数書き込み部 2 1 6 は、分析フィルタ処理実行部 2 1 1 より供給された出力係数を、係数並び替え用バッファ部 1 3 に書き込み、保持させる。

【 0 2 3 8 】

ステップ S 3 0 6 の処理が終了すると、制御部 2 0 1 は、処理をステップ S 3 0 2 に戻し、それ以降の処理を繰り返す。つまり、制御部 2 0 1 は、ステップ S 3 0 2 乃至ステップ S 3 0 6 の処理を繰り返すことにより、図 2 2 を参照して説明したように、入力された画像データに対応する、実行可能なリフティング演算を全て行わせる。

【 0 2 3 9 】

そして、ステップ S 3 0 5 において、実行可能なリフティング演算が無くなったと判定した場合、制御部 2 0 1 は、処理をステップ S 3 0 7 に進める。ステップ S 3 0 7 において、制御部 2 0 1 は、1 プレシント分のフィルタ処理が終了したか否かを判定し、終了していないと判定した場合、処理をステップ S 3 0 8 に進める。

【 0 2 4 0 】

ステップ S 3 0 8 において、制御部 2 0 1 は、分析フィルタ処理実行部 2 1 1 を制御し、リフティング演算により得られた出力係数を、係数並び替え用バッファ部 1 3 に保持させる。この制御により、分析フィルタ処理実行部 2 1 1 は、出力係数書き込み部 2 1 6 に出力係数を供給するとともに、その出力係数書き込み部 2 1 6 を制御し、供給した出力係数を係数並び替え用バッファ部 1 3 に書き込ませる。この制御により、出力係数書き込み部 2 1 6 は、分析フィルタ処理実行部 2 1 1 より供給された出力係数を、係数並び替え用バッファ部 1 3 に書き込み、保持させる。

【 0 2 4 1 】

ステップ S 3 0 9 において、制御部 2 0 1 は、係数並び替え用バッファ部 1 3 に保持されている現在の処理対象プレシントの係数データより、処理対象出力係数群を特定する。つまり、制御部 2 0 1 は、次のリフティング演算開始タイミングまでの空き時間に符号化する出力係数を特定する。

【 0 2 4 2 】

ステップ S 3 1 0 において、制御部 2 0 1 は、係数並び替え処理実行部 2 1 2 を制御し、処理対象出力係数として指定された係数データを所定順に読み出させる。この制御により、係数並び替え処理実行部 2 1 2 は、出力係数読み出し部 2 1 7 を制御し、係数並び替え用バッファ部 1 3 に保持されている係数データ（処理対象出力係数）を、合成フィルタ処理を行う順に読み出させて供給させる。この制御により、出力係数読み出し部 2 1 7 は、係数並び替え用バッファ部 1 3 に保持されている係数データを、合成フィルタ処理を行う順に読み出し、読み出した係数データを係数並び替え処理実行部 2 1 2 に供給させる。係数並び替え処理実行部 2 1 2 は、供給された係数データを順次エントロピ符号化処理実行部 2 1 3 に供給する。

10

20

30

40

50

【 0 2 4 3 】

ステップ S 3 1 1 において、制御部 2 0 1 は、エントロピ符号化処理実行部 2 1 3 を制御し、係数並び替え用バッファ部 1 3 より読み出された係数データをエントロピ符号化させる。この制御により、エントロピ符号化処理実行部 2 1 3 は、読み出された係数データを符号化する。

【 0 2 4 4 】

ステップ S 3 1 2 において、制御部 2 0 1 は、エントロピ符号化処理実行部 2 1 3 を制御し、ステップ S 3 1 1 において生成された符号化データをコードバッファ部 3 1 1 に保持させる。この制御により、エントロピ符号化処理実行部 2 1 3 は、符号化データ書き込み部 3 1 2 に符号化データを供給するとともに、符号化データ書き込み部 3 1 2 を制御し、供給した符号化データをコードバッファ部 3 1 1 に書き込ませる。この制御により、符号化データ書き込み部 3 1 2 は、エントロピ符号化処理実行部 2 1 3 より供給された出力係数を、コードバッファ部 3 1 1 に書き込み、保持させる。

10

【 0 2 4 5 】

ステップ S 3 1 3 において、制御部 2 0 1 は、係数並び替え用バッファ部 1 3 に、符号化処理していない処理対象出力係数が存在するか否かを判定し、存在すると判定した場合、処理をステップ S 3 1 0 に戻し、それ以降の処理を繰り返す。つまり、制御部 2 0 1 は、ステップ S 3 1 0 乃至ステップ S 3 1 3 の処理を繰り返すことにより、係数並び替え用バッファ部 1 3 に保持されている全ての処理対象出力係数に対して符号化処理を実行させる。

20

【 0 2 4 6 】

ステップ S 3 1 3 において、係数並び替え用バッファ部 1 3 に未処理の処理対象出力係数が存在しないと判定した場合、制御部 2 0 1 は、処理をステップ S 3 0 1 に戻し、それ以降の処理を繰り返す。つまり、制御部 2 0 1 は、ステップ S 3 0 1 乃至ステップ S 3 1 3 の処理を繰り返すことにより、ピクチャ内の全プレシнктについて、ウェーブレット変換処理を実行させ、さらにその空き時間にエントロピ符号化処理を実行させることにより、ウェーブレット変換処理とエントロピ符号化処理を見かけ上同時並行的に実行させる。

【 0 2 4 7 】

ステップ S 3 0 7 において、1 プレシнкт分のフィルタ処理が終了したと判定した場合、制御部 2 0 1 は、処理を図 2 6 のステップ S 3 2 1 に進める。

30

【 0 2 4 8 】

図 2 6 のステップ S 3 2 1 において、制御部 2 0 1 は、エントロピ符号化処理実行部 2 1 3 を制御し、最後のリフティング演算の出力係数をエントロピ符号化させる。つまり、この最後のリフティング演算の出力係数は、係数並び替え用バッファ部 1 3 を介さずにエントロピ符号化させる。

【 0 2 4 9 】

ステップ S 3 2 1 の制御により、エントロピ符号化処理実行部 2 1 3 は、例えば 5 番目に合成フィルタ処理される係数データおよび 6 番目に合成フィルタ処理される係数データを順に符号化し、得られた符号化データを順次出力する。

40

【 0 2 5 0 】

その符号化処理が終了すると、制御部 2 0 1 は、ステップ S 3 2 2 において、符号化データ読み出し部 3 1 3 を制御し、コードバッファ部 3 1 1 より符号化データを所定順に読み出させる。この制御により、符号化データ読み出し部 3 1 3 は、コードバッファ部 3 1 1 に保持されている符号化データを、例えば合成フィルタ処理順に読み出し、それをエントロピ符号化処理実行部 2 1 3 に供給する。エントロピ符号化処理実行部 2 1 3 は、その供給された符号化データを、ステップ S 3 2 1 において符号化して出力した符号化データに続けて出力する。

【 0 2 5 1 】

ステップ S 3 2 3 において、制御部 2 0 1 は、1 ピクチャ分のフィルタ処理が終了した

50

か否かを判定し、1ピクチャ分のフィルタ処理が終了して無いと判定した場合、処理を図25のステップS301に戻し、それ以降の処理を繰り返す。

【0252】

また、ステップS323において、1ピクチャ分のフィルタ処理が終了したと判定された場合、制御部201は、処理をステップS324に進め、係数並び替え用バッファ部13に保持されている残りの出力係数を処理対象出力係数群に設定する。

【0253】

制御部201は、ステップS325において、係数並び替え処理実行部212を制御し、処理対象出力係数として指定された係数データを所定順に読み出させる。この制御により、係数並び替え処理実行部212は、出力係数読み出し部217を制御し、係数並び替え用バッファ部13に保持されている係数データ（処理対象出力係数）を、合成フィルタ処理を行う順に読み出させて供給させる。この制御により、出力係数読み出し部217は、係数並び替え用バッファ部13に保持されている係数データを、合成フィルタ処理を行う順に読み出し、読み出した係数データを係数並び替え処理実行部212に供給させる。係数並び替え処理実行部212は、供給された係数データを順次エントロピ符号化処理実行部213に供給する。

10

【0254】

ステップS326において、制御部201は、エントロピ符号化処理実行部213を制御し、係数並び替え用バッファ部13より読み出された係数データをエントロピ符号化させる。この制御により、エントロピ符号化処理実行部213は、読み出された係数データを符号化し、得られた符号化データを出力する。

20

【0255】

ステップS327において、制御部201は、係数並び替え用バッファ部13に、符号化処理していない処理対象出力係数が存在するか否かを判定し、存在すると判定した場合、処理をステップS325に戻し、それ以降の処理を繰り返す。

【0256】

ステップS327において、係数並び替え用バッファ部13に未処理の処理対象出力係数が存在しないと判定した場合、制御部201は、制御処理を終了する。

【0257】

以上のように制御処理を行うことにより、制御部201は、ウェーブレット変換処理とエントロピ符号化処理を同時並列的に実行し、より容易に、所定の速度で入力される画像データよりリアルタイムに符号化データを生成して出力することができる。また、これにより、保持するデータを符号化データにすることができるので、処理に必要なメモリ量を低減させ、ウェーブレット変換処理および符号化処理の負荷を低減させることができる。

30

【0258】

なお、図25および図26においては、制御処理の流れの例を説明したが、分析フィルタ処理実行部211乃至出力係数読み出し部217は、上述したように、この制御処理の各ステップに対応して制御部201の制御に対応する処理を実行する。従って、図25および図26のフローチャートに示される制御処理の各ステップは、上述したとおりに、分析フィルタ処理実行部211乃至符号化データ読み出し部313のそれぞれが実行する処理も表しているものとし、各部が実行する処理の図示は省略する。

40

【0259】

以上においては、図10を参照して説明したように、処理演算部としてのCPUが1つであるパーソナルコンピュータ100においてソフトウェアプログラムを実行する場合について説明したが、近年においては、CPUを複数有する情報処理装置も多く存在する。また、情報処理装置がCPUを1つしか有していなくても、そのCPUが、互いに独立して動作することができるコアを複数有する場合もある。このように、互いに独立して動作する複数の演算処理部を有する情報処理装置は、複数の処理を互いに並行して実行することができる。以下においては、このような互いに独立して動作する複数の演算処理部を有する情報処理装置を用いて、上述したようなソフトウェアエンコーダを実現させる場合について説明

50

する。

【 0 2 6 0 】

図 2 7 は、パーソナルコンピュータの他の構成例を示すブロック図である。図 2 7 に示されるパーソナルコンピュータ 4 0 0 は、基本的に図 1 0 に示されるパーソナルコンピュータ 1 0 0 と同様の構成を有する。すなわち、パーソナルコンピュータ 4 0 0 は、パーソナルコンピュータ 1 0 0 の ROM 1 0 2 と同様の ROM 4 0 3、パーソナルコンピュータ 1 0 0 の RAM 1 0 3 と同様の RAM 4 0 4、パーソナルコンピュータ 1 0 0 のバス 1 0 4 と同様のバス 4 0 5、パーソナルコンピュータ 1 0 0 の入出力インタフェース 1 1 0 と同様の入出力インタフェース 4 1 0、パーソナルコンピュータ 1 0 0 の入力部 1 1 1 と同様の入力部 4 1 1、パーソナルコンピュータ 1 0 0 の出力部 1 1 2 と同様の出力部 4 1 2、パーソナルコンピュータ 1 0 0 の記憶部 1 1 3 と同様の記憶部 4 1 3、パーソナルコンピュータ 1 0 0 の通信部 1 1 4 と同様の通信部 4 1 4、および、パーソナルコンピュータ 1 0 0 のドライブ 1 1 5 と同様のドライブ 4 1 5 を有する。ドライブ 4 1 5 には、パーソナルコンピュータ 1 0 0 のリムーバブルメディア 1 2 1 と同様のリムーバブルメディア 4 2 1 が適宜装着される。

10

【 0 2 6 1 】

しかしながら、パーソナルコンピュータ 4 0 0 は、パーソナルコンピュータ 1 0 0 が 1 つの CPU 1 0 1 を有するのとは異なり、CPU 4 0 1 および CPU 4 0 2 を有する。CPU 4 0 1 および CPU 4 0 2 のそれぞれは、基本的に CPU 1 0 1 と同様である。CPU 4 0 1 と CPU 4 0 2 は、互いに独立して処理を実行することができる。すなわち、パーソナルコンピュータ 4 0 0 は、互いに独立して動作する 2 つの演算処理部 (CPU 4 0 1 および CPU 4 0 2) を有する。

20

【 0 2 6 2 】

このようなパーソナルコンピュータ 4 0 0 において図 1 の符号化部 1 0 を実現させる場合、CPU 4 0 1 と CPU 4 0 2 とを使用して、図 2 8 のように処理が実行される。図 2 8 に、CPU 4 0 1 および CPU 4 0 2 において実行される処理の流れを模式的に示す。図 2 8 は、図 1 2 に対応する図であり、基本的には図 1 2 と同様の構成であるので、その詳細についての説明は省略する。

【 0 2 6 3 】

図 2 8 の例の場合、互いに独立して動作する 2 つの演算処理部 (CPU 4 0 1 および CPU 4 0 2) が存在するので、一方の演算処理部である CPU 4 0 1 において、ウェーブレット変換処理を実行させ、他方の演算処理部である CPU 4 0 2 において、エントロピ符号化処理を実行させる。このようにすることにより、ウェーブレット変換処理とエントロピ符号化処理を互いに並行して実行することができる。図 2 8 の例の場合、上述した例と異なり、時分割による見かけ上の並列化ではなく、2 つの演算処理部を用いて 2 つの処理を実際に並行して動作させる。

30

【 0 2 6 4 】

上述した他の例と同様に、一方の CPU 4 0 1 は、画像データが 2 ライン入力される毎に実行可能なリフティング演算 (DWT) を行い、出力係数を生成して係数並び替え用バッファ部 1 3 に保持させる。この分析フィルタ処理の実行中に、他方の CPU 4 0 2 は、この分析フィルタ処理と並行して、係数並び替え用バッファ部 1 3 に蓄積されている出力係数を、合成フィルタ処理を行う順に読み出してエントロピ符号化 (VLC) し、得られた符号化データを順次出力する。

40

【 0 2 6 5 】

このように 2 つの演算処理部を用いることにより、各 CPU の負荷が低減される。従って、上述した他の例の場合よりも低速に動作する CPU であってもオーバーフローせずに、所定の速度で入力される画像データに対して、ウェーブレット変換処理とエントロピ符号化処理を容易にリアルタイムに実行することができる。

【 0 2 6 6 】

また、図 2 9 に示されるように、この場合、CPU 4 0 1 が第 2 プレシントに対してウェーブレット変換 (P2 DWT) を行っているのと並行して、CPU 4 0 2 が第 1 プレシント

50

に対してエントロピ符号化を行う(P1VLC)。従って、この場合の遅延時間は、上述した他の例と略同じである。また、CPU402は、符号化処理のみを行うので、符号化データの出力タイミングを適宜分散させることができる。従って、符号化部10の後段の処理が容易になる。

【0267】

また、図28に示されるように、CPU401が、出力係数を係数並び替え用バッファ部13に保持させるのと並行してCPU402が出力係数を読み出す。従って、係数並び替え用バッファ部13に同時に保持されるデータ量は、上述した他の例より少なくなる。つまり、この場合、係数並び替え用バッファ部13として必要なメモリ容量を低減させることができる。

10

【0268】

ただし、この場合、CPU401とCPU402とで動作を連携させる必要があり、CPU401において実行される処理と、CPU402において実行される処理の両方を制御する仕組みが必要になる。

【0269】

図30は、上述した流れで処理を実行する符号化部10を実現するソフトウェアプログラムを実行するCPU401およびCPU402が有する機能を示す機能ブロック図である。

【0270】

図30も、図14に対応する図であり、図14に示されるブロックと対応するブロックには同一の番号を付してある。図30の場合、図14や図20に示される分析フィルタ処理実行部211乃至出力係数読み出し部217の各処理部は、CPU401またはCPU402に割り当てられる。

20

【0271】

CPU401は、分析フィルタ処理に関する機能を有しており、分析フィルタ処理実行部211、途中計算用データ読み出し部214、途中計算用データ書き込み部215、および出力係数書き込み部216を有する。CPU402は、係数並び替えおよびエントロピ符号化に関する機能を有しており、係数並び替え処理実行部212、エントロピ符号化処理実行部213、および出力係数読み出し部217を有する。

【0272】

なお、図30の例においては、途中計算用バッファ部12がCPU401のキャッシュメモリ(図示せず)に形成され、係数並び替え用バッファ部13がRAM404に形成されるものとして説明を行うが、上述したように、途中計算用バッファ部12および係数並び替え用バッファ部13は、任意の記憶領域に形成することができる。ただし、アクセス頻度の高い途中計算用バッファ部12は、負荷や遅延時間を低減させるために、分析フィルタ処理が行われるCPU401のキャッシュメモリに形成させるのが望ましい。

30

【0273】

図30に示されるように、CPU401およびCPU402の両方の機能ブロックにより、上述した他の例のCPU101と同様に、図1の符号化部10を実現する。つまり、CPU401において、画像データに対して分析フィルタ処理が行われ、その出力係数が係数並び替え用バッファ部13に保持され、CPU402において、その出力係数を合成フィルタ処理を行う順に読み出し、各係数に対して符号化処理を行い、符号化データを生成して出力する。

40

【0274】

CPU401は、さらに、制御部511および分析フィルタ制御部512を有し、CPU402は、さらに、並び替え符号化制御部513を有する。

【0275】

制御部511は、CPU401の分析フィルタ制御部512およびCPU402の並び替え符号化制御部513の動作を制御する。分析フィルタ制御部512は、CPU401が有する各機能ブロックの動作を制御し、CPU401において実行される分析フィルタ処理に関する制御を行う。並び替え符号化制御部513は、CPU402が有する各機能ブロックの動

50

作を制御し、CPU 4 0 2 において実行される係数の並び替えやエントロピ符号化に関する制御を行う。

【 0 2 7 6 】

図 3 1 のフローチャートを参照して、制御部 5 1 1 が実行する制御処理の流れの例を説明する。なお、この制御処理は、入力される画像データのピクチャ毎に実行される。

【 0 2 7 7 】

制御処理が開始されると、制御部 5 1 1 は、ステップ S 4 0 1 において、分析フィルタ制御部 5 1 2 を制御し、分析フィルタ制御処理を開始させる。この制御により、分析フィルタ制御部 5 1 2 は、分析フィルタ制御処理を開始する。分析フィルタ制御処理については後述する。ステップ S 4 0 2 において、制御部 5 1 1 は、並び替え符号化制御部 5 1 3 を制御し、並び替え符号化制御処理を開始させる。この制御により、並び替え符号化制御部 5 1 3 は、並び替え符号化制御処理を開始する。並び替え符号化制御処理については後述する。

10

【 0 2 7 8 】

図 3 2 のフローチャートを参照して、分析フィルタ制御部 5 1 2 が実行する分析フィルタ制御処理の流れの例を説明する。

【 0 2 7 9 】

分析フィルタ制御処理が開始されると、分析フィルタ制御部 5 1 2 は、ステップ S 4 1 1 において、所定ライン数（例えば 2 ラインまたは 3 ライン）分の画像データが入力されたか否かを判定し、入力されたと判定するまで待機する。そして、入力されたと判定した場合、分析フィルタ制御部 5 1 2 は、処理をステップ S 4 1 2 に進める。

20

【 0 2 8 0 】

ステップ S 4 1 2 において、分析フィルタ制御部 5 1 2 は、分析フィルタ処理実行部 2 1 1 を制御し、途中計算用バッファ部 1 2 より、これから行うリフティング演算に必要な途中計算用データを取得させる。この制御により、分析フィルタ処理実行部 2 1 1 は、途中計算用データ読み出し部 2 1 4 を制御して、これから行うリフティング演算に必要な途中計算用データを、途中計算用バッファ部 1 2 より読み出させ、供給させる（つまり、途中計算用データを取得する）。この制御により、途中計算用データ読み出し部 2 1 4 は、要求された途中計算用データを途中計算用バッファ部 1 2 より読み出し、分析フィルタ処理実行部 2 1 1 に供給する。

30

【 0 2 8 1 】

ステップ S 4 1 3 において、分析フィルタ制御部 5 1 2 は、分析フィルタ処理実行部 2 1 1 を制御し、リフティング演算 1 回分の分析フィルタ処理を実行させる。この制御により、分析フィルタ処理実行部 2 1 1 は、図 2 8 を参照して説明したように、実行可能な分析フィルタ処理のリフティング演算を 1 回分実行する。

【 0 2 8 2 】

リフティング演算が行われると、分析フィルタ制御部 5 1 2 は、ステップ S 4 1 4 において、分析フィルタ処理実行部 2 1 1 を制御し、リフティング演算に用いられた画像データや係数データ、およびリフティング演算により得られた係数データのうち、次回以降のリフティング演算に利用するデータを途中計算用データとして途中計算用バッファ部 1 2 に保持させる。この制御により、分析フィルタ処理実行部 2 1 1 は、途中計算用データ書き込み部 2 1 5 に途中計算用データを供給するとともに、その途中計算用データ書き込み部 2 1 5 を制御し、供給した途中計算用データを途中計算用バッファ部 1 2 に書き込ませる。この制御により、途中計算用データ書き込み部 2 1 5 は、分析フィルタ処理実行部 2 1 1 より供給された途中計算用データを、途中計算用バッファ部 1 2 に書き込み、保持させる。

40

【 0 2 8 3 】

ステップ S 4 1 5 において、分析フィルタ制御部 5 1 2 は、分析フィルタ処理実行部 2 1 1 を制御し、リフティング演算により得られた出力係数を、係数並び替え用バッファ部 1 3 に保持させる。この制御により、分析フィルタ処理実行部 2 1 1 は、出力係数書き込

50

み部 2 1 6 に出力係数を供給するとともに、その出力係数書き込み部 2 1 6 を制御し、供給した出力係数を係数並び替え用バッファ部 1 3 に書き込ませる。この制御により、出力係数書き込み部 2 1 6 は、分析フィルタ処理実行部 2 1 1 より供給された出力係数を、係数並び替え用バッファ部 1 3 に書き込み、保持させる。

【 0 2 8 4 】

ステップ S 4 1 6 において、分析フィルタ制御部 5 1 2 は、新たな画像データの入力無しに、さらにリフティング演算が可能であるか否かを判定し、可能であると判定した場合、処理をステップ S 4 1 2 に戻し、それ以降の処理を繰り返す。つまり、分析フィルタ制御部 5 1 2 は、ステップ S 4 1 2 乃至ステップ S 4 1 6 の処理を繰り返すことにより、図 2 8 を参照して説明したように、入力された画像データに対応する、実行可能なリフティング演算を全て行わせる。

10

【 0 2 8 5 】

そして、ステップ S 4 1 6 において、実行可能なリフティング演算が無くなったと判定した場合、分析フィルタ制御部 5 1 2 は、処理をステップ S 4 1 7 に進める。ステップ S 4 1 7 において、分析フィルタ制御部 5 1 2 は、1 ピクチャ分のフィルタ処理が終了したか否かを判定し、1 ピクチャ分のフィルタ処理が終了して無いと判定した場合、処理をステップ S 4 1 1 に戻し、それ以降の処理を繰り返す。

【 0 2 8 6 】

また、ステップ S 4 1 7 において、1 ピクチャ分のフィルタ処理が終了したと判定された場合、分析フィルタ制御部 5 1 2 は、分析フィルタ制御処理を終了する。

20

【 0 2 8 7 】

次に、図 3 3 のフローチャートを参照して、並び替え符号化制御部 5 1 3 が実行する並び替え符号化制御処理の流れの例を説明する。

【 0 2 8 8 】

並び替え符号化制御処理が開始されると、並び替え符号化制御部 5 1 3 は、ステップ S 4 2 1 において、係数並び替え用バッファ部 1 3 に、符号化処理していない出力係数が存在するか否かを判定し、存在すると判定した場合、処理をステップ S 4 2 2 に進める。ステップ S 4 2 2 において、並び替え符号化制御部 5 1 3 は、係数並び替え処理実行部 2 1 2 を制御し、係数並び替え用バッファ部 1 3 に保持されている出力係数を所定順に読み出させる。この制御により、係数並び替え処理実行部 2 1 2 は、出力係数読み出し部 2 1 7 を制御し、係数並び替え用バッファ部 1 3 に保持されている係数データ（出力係数）を、合成フィルタ処理を行う順に読み出させて供給させる。この制御により、出力係数読み出し部 2 1 7 は、係数並び替え用バッファ部 1 3 に保持されている係数データを、合成フィルタ処理を行う順に読み出し、読み出した係数データを係数並び替え処理実行部 2 1 2 に供給させる。係数並び替え処理実行部 2 1 2 は、供給された係数データを順次エントロピ符号化処理実行部 2 1 3 に供給する。

30

【 0 2 8 9 】

ステップ S 4 2 3 において、並び替え符号化制御部 5 1 3 は、エントロピ符号化処理実行部 2 1 3 を制御し、係数並び替え用バッファ部 1 3 より読み出された係数データをエントロピ符号化させる。この制御により、エントロピ符号化処理実行部 2 1 3 は、読み出された係数データを符号化し、得られた符号化データを出力する。

40

【 0 2 9 0 】

ステップ S 4 2 3 の処理を終了すると、並び替え符号化制御部 5 1 3 は、処理をステップ S 4 2 4 に進める。また、ステップ S 4 2 1 において、未処理の出力係数が存在しないと判定した場合、並び替え符号化制御部 5 1 3 は、ステップ S 4 2 2 およびステップ S 4 2 3 の処理を省略し、処理をステップ S 4 2 4 に進める。

【 0 2 9 1 】

ステップ S 4 2 4 において、並び替え符号化制御部 5 1 3 は、1 ピクチャ分のフィルタ処理が終了したか否かを判定し、1 ピクチャ分のフィルタ処理が終了していないと判定した場合、処理をステップ S 4 2 1 に戻し、それ以降の処理を繰り返す。

50

【 0 2 9 2 】

また、ステップ S 4 2 4 において、1 ピクチャ分の フィルタ処理 が終了したと判定された場合、並び替え符号化制御部 5 1 3 は、並び替え符号化制御処理を終了する。

【 0 2 9 3 】

以上のように各制御処理を行うことにより、制御部 5 1 1、分析フィルタ制御部 5 1 2、および並び替え符号化制御部 5 1 3 は、ウェーブレット変換処理とエントロピ符号化処理を、2つの演算処理部を利用して実際に並行して実行し、容易に、所定の速度で入力される画像データよりリアルタイムに符号化データを生成して出力することができる。

【 0 2 9 4 】

なお、図 2 7 においては、CPUの個数が2つであるように説明したが、上述した処理は、3つ以上のCPUを有するパーソナルコンピュータにおいても実行可能である。また、図 2 6 より前の図面を参照して説明した他の制御方法を、図 2 7 に示されるパーソナルコンピュータ 4 0 0 のように複数の演算処理部を有する情報処理装置において実行することもできる。

10

【 0 2 9 5 】

また、以上に説明した各実施の形態においては、それぞれの制御部が、分析フィルタ処理実行部 2 1 1 乃至出力係数読み出し部 2 1 7 等の各処理部を制御するように説明したが、上述した制御処理は、分析フィルタ処理実行部 2 1 1 乃至出力係数読み出し部 2 1 7 等の各処理部において行われるようにしてもよい。つまり、制御部を別途設けずに、実際にデータを処理する分析フィルタ処理実行部 2 1 1 乃至出力係数読み出し部 2 1 7 等の各処理部が適宜互いを制御しあうようにしてもよい。

20

【 0 2 9 6 】

以上においては、ウェーブレット変換処理をリフティング演算処理 1 回分毎に制御し、また、符号化処理を各係数データに対する処理毎に制御するように説明したが、ウェーブレット変換処理および符号化処理のそれぞれを複数の工程に分割する処理単位であれば、どのような処理単位で制御するようにしてもよい。例えば、ウェーブレット変換処理をリフティング演算処理 1 回分よりもさらに細かい処理単位で制御するようにし、より細かな制御を行いCPUの使用効率をさらに向上させることができるようにしてもよいし、複数のリフティング演算処理を処理単位とし、制御処理をより簡易化するようにしてもよい。また、例えば、符号化処理を係数データ毎の処理よりさらに細かい処理単位で制御するようにし、より細かな制御を行いCPUの使用効率をさらに向上させることができるようにしてもよいし、複数の係数データに対する処理を処理単位とし、制御処理をより簡易化するようにしてもよい。

30

【 0 2 9 7 】

また、以上においては分析フィルタ処理に 5×3 フィルタを用いるように説明したが、フィルタ処理に使用されるフィルタはどのようなフィルタを用いてもよく、例えば、 9×7 フィルタを用いるようにしてもよい。

【 0 2 9 8 】

上述した一連の処理は、ソフトウェアにより実行させることもできるが、ハードウェアにより実行させることもできる。一連の処理をソフトウェアにより実行させる場合には、そのソフトウェアを構成するプログラムが、専用のハードウェアに組み込まれているコンピュータ、または、各種のプログラムをインストールすることで、各種の機能を実行することが可能な、例えば汎用のパーソナルコンピュータ、または、複数の装置よりなる情報処理システムの情報処理装置などに、プログラム記録媒体からインストールされる。

40

【 0 2 9 9 】

この記録媒体は、例えば、図 1 0 や図 2 7 に示されるように、装置本体とは別に、ユーザにプログラムを配信するために配布される、プログラムが記録されている磁気ディスク（フレキシブルディスクを含む）、光ディスク（CD-ROM(Compact Disk-Read Only Memory)、DVD(Digital Versatile Disk)を含む）、光磁気ディスク（MD (Mini-Disk)（登録商標）を含む）、もしくは半導体メモリなどよりなるリムーバブルメディア 1 2 1 またはリム

50

ーバブルメディア 4 2 1 により構成されるだけでなく、装置本体に予め組み込まれた状態でユーザに配信される、プログラムが記録されているROM 1 0 2 若しくはROM 4 0 3、または、記憶部 1 1 3 若しくは記憶部 4 1 3 などで構成される。

【 0 3 0 0 】

なお、本明細書において、記録媒体に記録されるプログラムを記述するステップは、記載された順序に沿って時系列的に行われる処理はもちろん、必ずしも時系列的に処理されなくとも、並列的あるいは個別に実行される処理をも含むものである。

【 0 3 0 1 】

また、本明細書において、システムとは、複数のデバイス（装置）により構成される装置全体を表すものである。

【 0 3 0 2 】

なお、以上において、1つの装置として説明した構成を分割し、複数の装置として構成するようにしてもよい。逆に、以上において複数の装置として説明した構成をまとめて1つの装置として構成されるようにしてもよい。また、各装置の構成に上述した以外の構成を付加するようにしてももちろんよい。さらに、システム全体としての構成や動作が実質的に同じであれば、ある装置の構成の一部を他の装置の構成に含めるようにしてもよい。つまり、本発明の実施の形態は、上述した実施の形態に限定されるものではなく、本発明の要旨を逸脱しない範囲において種々の変更が可能である。

【産業上の利用可能性】

【 0 3 0 3 】

以上説明した本発明は、複数の端末間で画像を圧縮して伝送、受信して伸長して画像を出力するシステム、特に低遅延で実現する必要性が高いプロダクツ・アプリケーションに好適である。具体的には、放送局で多用するデジタルトリアックスシステム、ライブ映像配信システム、TV会議システム、監視カメラ・レコーダシステム、医用遠隔医療診断、生徒と教師間のインタラクティブ通信、ビデオゲーム等に応用が可能である。また、画像データを圧縮符号化して記録媒体に記録する装置（記録再生装置も含む）やシステム等にも適用可能である。

【図面の簡単な説明】

【 0 3 0 4 】

【図 1】符号化部の構成例を示す図である。

【図 2】復号部の構成例を示す図である。

【図 3】ウェーブレット変換について概略的に説明するための略線図である。

【図 4】ウェーブレット変換について概略的に説明するための略線図である。

【図 5】ウェーブレット変換について概略的に説明するための略線図である。

【図 6】5 × 3 分析フィルタのリフティング構成の例を説明する図である。

【図 7】5 × 3 合成フィルタのリフティング構成の例を説明する図である。

【図 8】分析フィルタ処理および合成フィルタ処理の流れの例を説明する図である。

【図 9】符号化側および復号側の各処理が並列的に実行される様子を説明するための略線図である。

【図 10】パーソナルコンピュータの主な構成例を示すブロック図である。

【図 11】ウェーブレット変換処理とエントロピ符号化処理の流れの関係の例を説明する図である。

【図 12】CPUにおいて実行される処理の流れの例を説明する模式図である。

【図 13】CPUにおいて実行される処理の流れの例を説明する模式図である。

【図 14】CPUが有する機能の構成例を説明する機能ブロック図である。

【図 15】制御処理の流れの例を説明するフローチャートである。

【図 16】制御処理の流れの例を説明する、図 15 に続くフローチャートである。

【図 17】ウェーブレット変換処理とエントロピ符号化処理の流れの関係の他の例を説明する図である。

【図 18】CPUにおいて実行される処理の流れの、他の例を説明する模式図である。

10

20

30

40

50

【図 19】CPUにおいて実行される処理の流れの、他の例を説明する模式図である。

【図 20】CPUが有する機能の、他の構成例を説明する機能ブロック図である。

【図 21】制御処理の流れの、他の例を説明するフローチャートである。

【図 22】CPUにおいて実行される処理の流れの、さらに他の例を説明する模式図である。

。

【図 23】CPUにおいて実行される処理の流れの、さらに他の例を説明する模式図である。

。

【図 24】CPUが有する機能の、さらに他の構成例を説明する機能ブロック図である。

【図 25】制御処理の流れの、さらに他の例を説明するフローチャートである。

【図 26】制御処理の流れの、さらに他の例を説明する、図 25 に続くフローチャートである。 10

【図 27】パーソナルコンピュータの、他の主な構成例を示すブロック図である。

【図 28】CPUにおいて実行される処理の流れの、さらに他の例を説明する模式図である。

。

【図 29】CPUにおいて実行される処理の流れの、さらに他の例を説明する模式図である。

。

【図 30】CPUが有する機能の、さらに他の構成例を説明する機能ブロック図である。

【図 31】制御処理の流れの、さらに他の例を説明するフローチャートである。

【図 32】分析フィルタ制御制御処理の流れの例を説明するフローチャートである。

【図 33】並び替え符号化制御処理の流れの例を説明するフローチャートである。 20

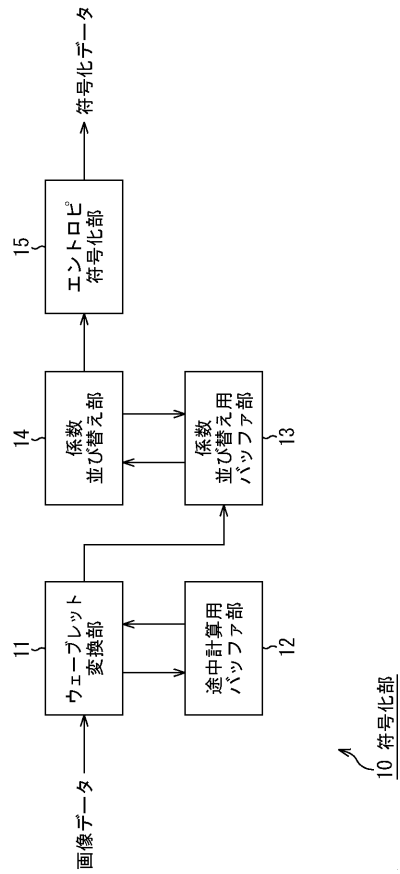
【符号の説明】

【0305】

10 符号化部, 11 ウェーブレット変換部, 12 途中計算用バッファ部,
13 係数並び替え用バッファ部, 14 係数並び替え部, 15 エントロピ符号化
部, 100 パーソナルコンピュータ, 101 CPU, 201 制御部, 211
分析フィルタ処理実行部, 212 係数並び替え処理実行部, 213 エントロピ
符号化処理実行部, 214 途中計算用データ読み出し部, 215 途中計算用データ
書き込み部, 216 出力係数書き込み部, 217 出力係数読み出し部, 22
1 画像データバッファ部, 311 コードバッファ部, 312 符号化データ書き
込み部, 313 符号化データ読み出し部, 401および402 CPU, 511 30
制御部, 512 分析フィルタ制御部, 513 並び替え符号化制御部

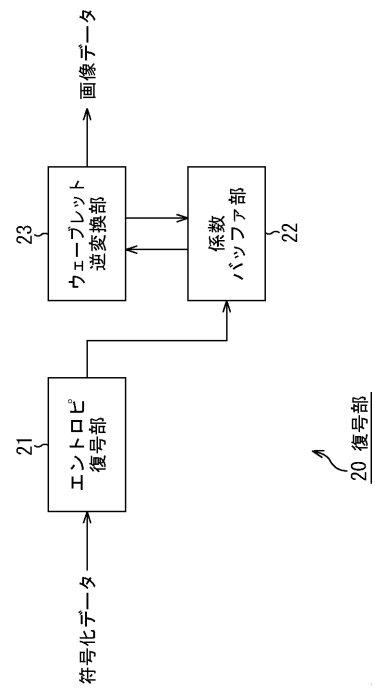
【 図 1 】

図1



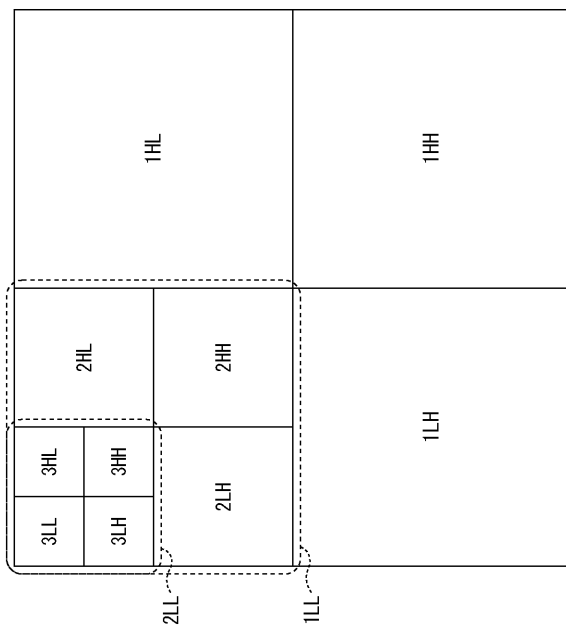
【 図 2 】

図2



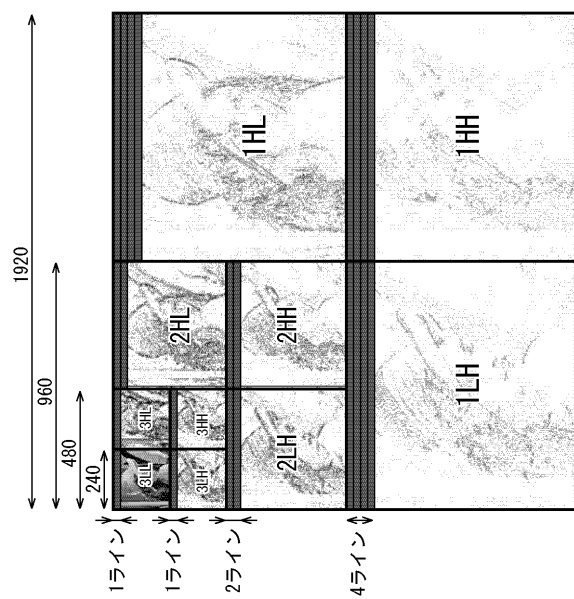
【 図 3 】

图3

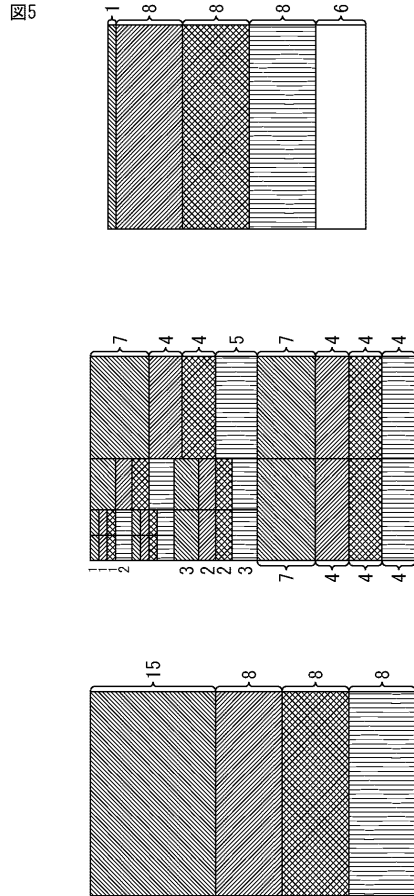


【圖 4】

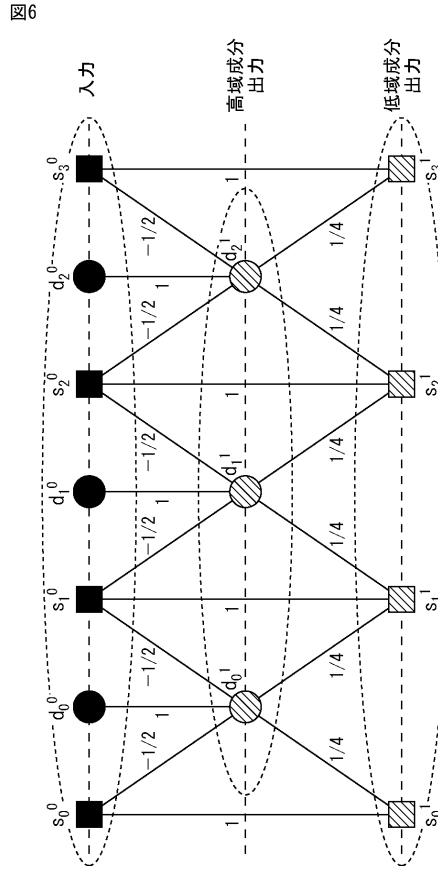
图4



【図5】



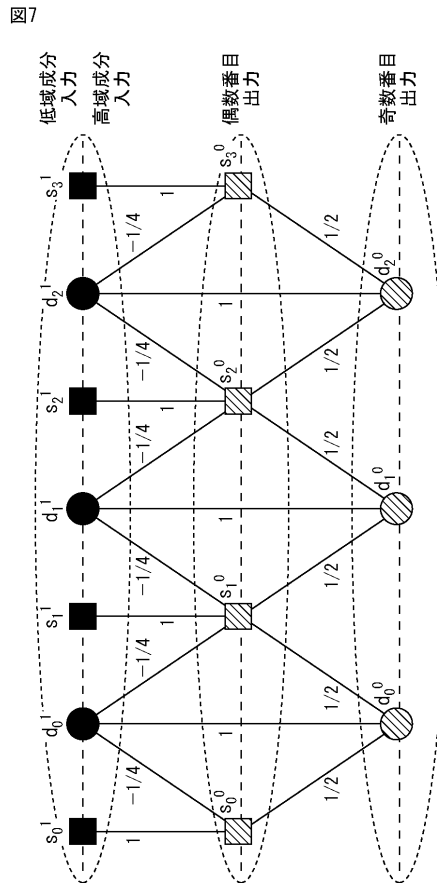
【図6】



$$d_1^1 = d_1^0 - 1/2(s_1^0 + s_{1+1}^0) : \text{高域成分出力}$$

$$s_1^1 = s_1^0 + 1/4(d_{1-1}^1 + d_1^1) : \text{低域成分出力}$$

【図7】



$$s_1^0 = s_1^1 - 1/4(d_{1-1}^1 + d_1^1) : \text{偶数番目出力}$$

$$d_1^0 = d_1^1 + 1/2(s_1^0 + s_{1+1}^0) : \text{奇数番目出力}$$

【図8】

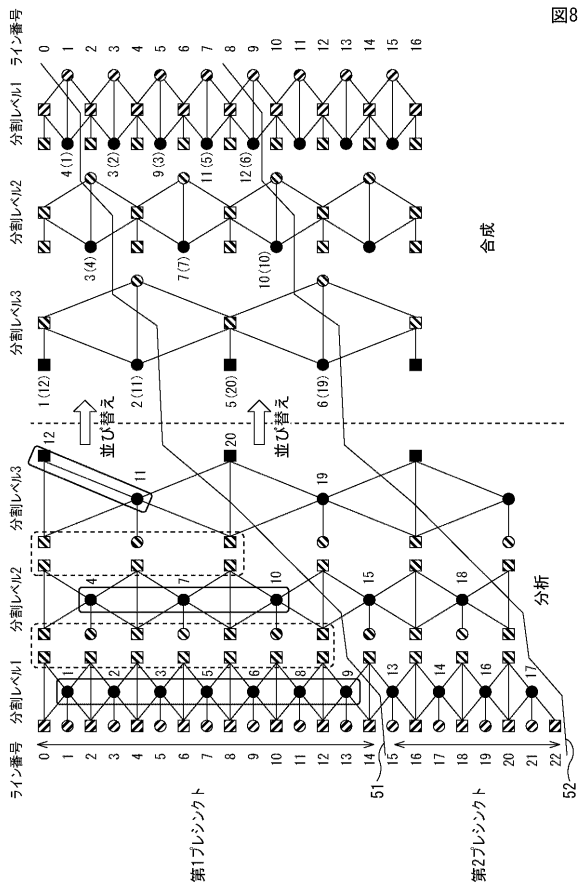
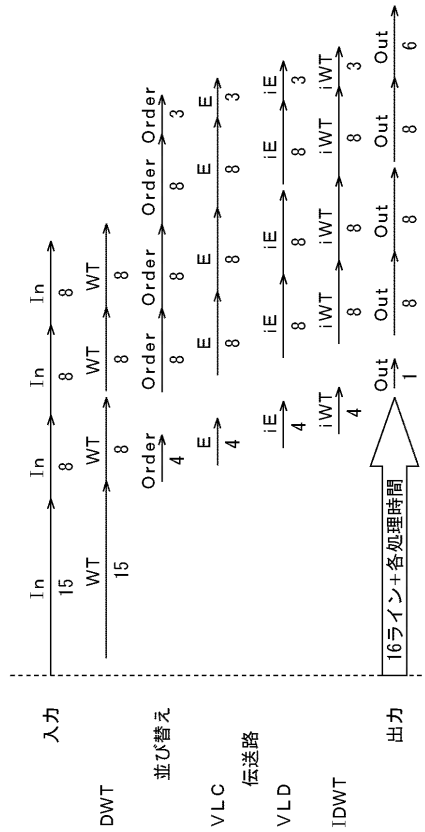


図8

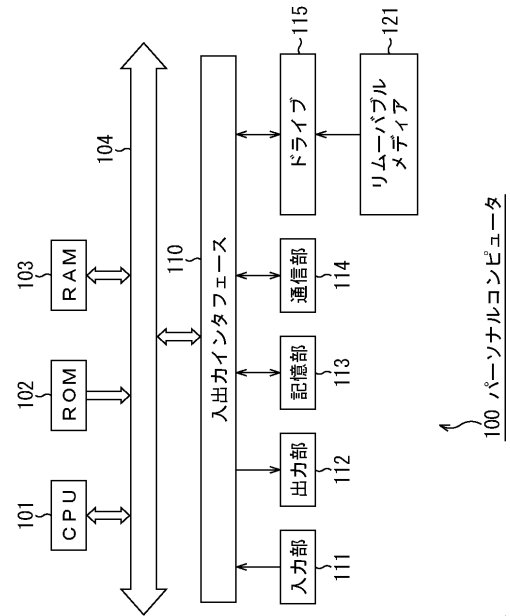
【 図 9 】

图9



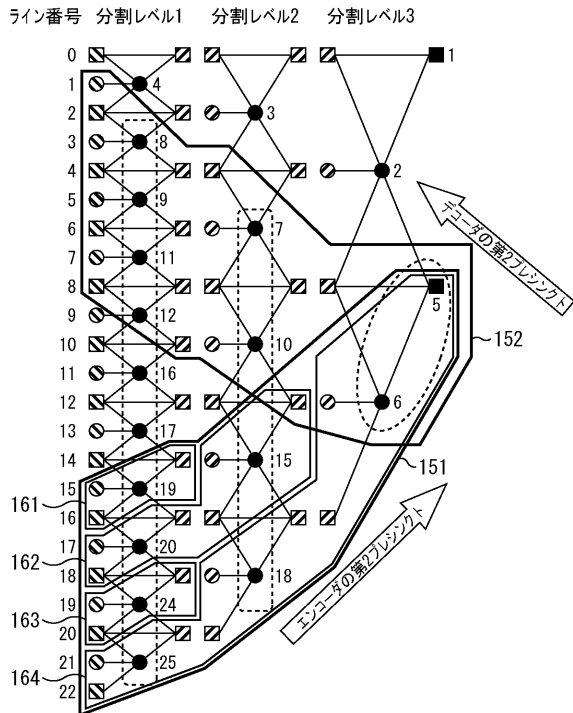
【 図 1 0 】

図10



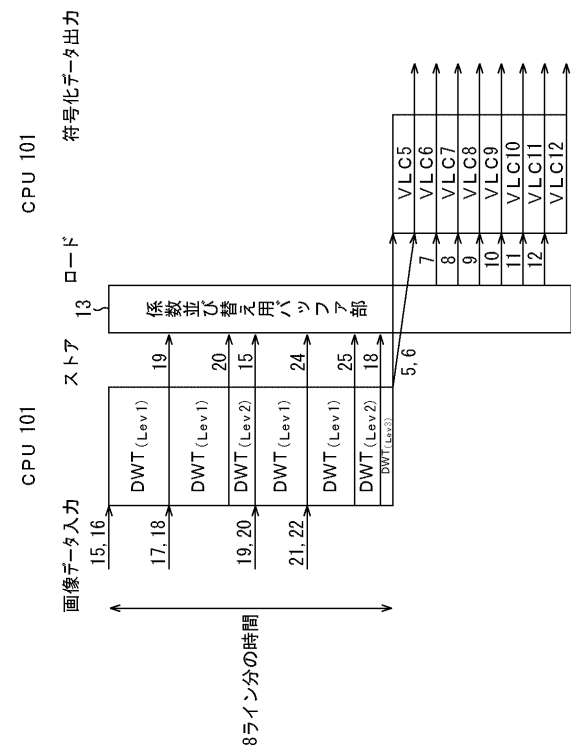
【 ㄨ 1 1 】

图11



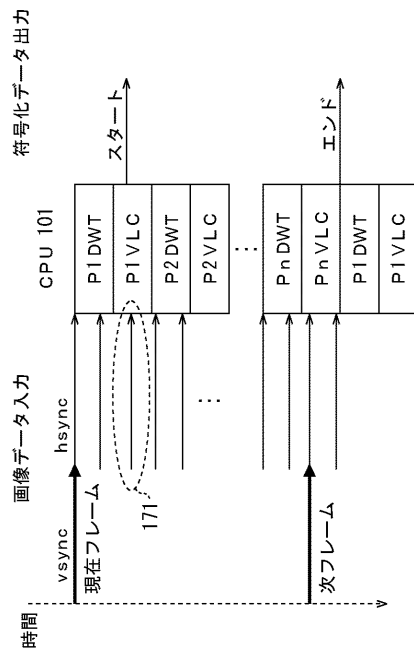
【 図 1 2 】

图12



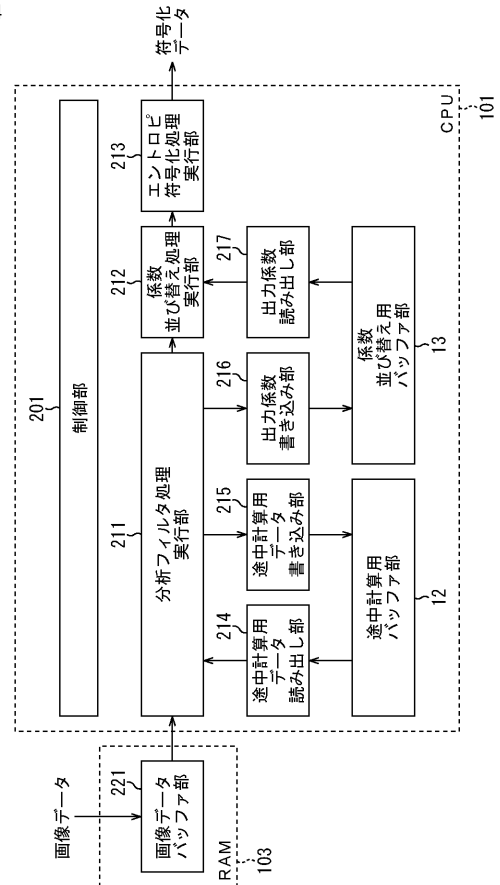
【 図 1 3 】

图13



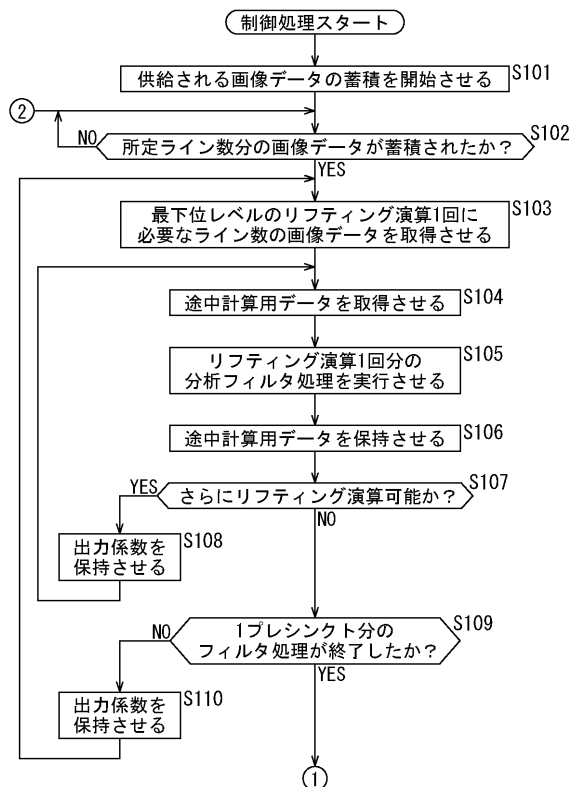
【 図 1 4 】

图14



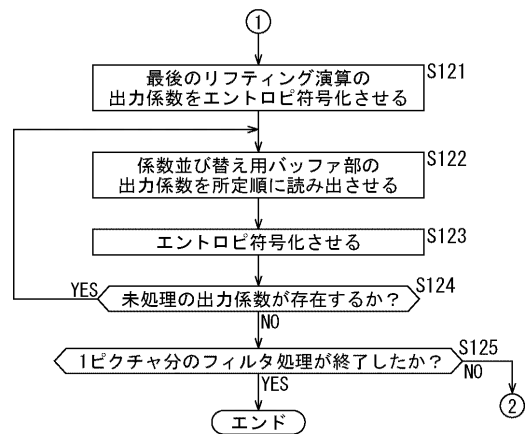
【 図 1 5 】

図15



【 図 1 6 】

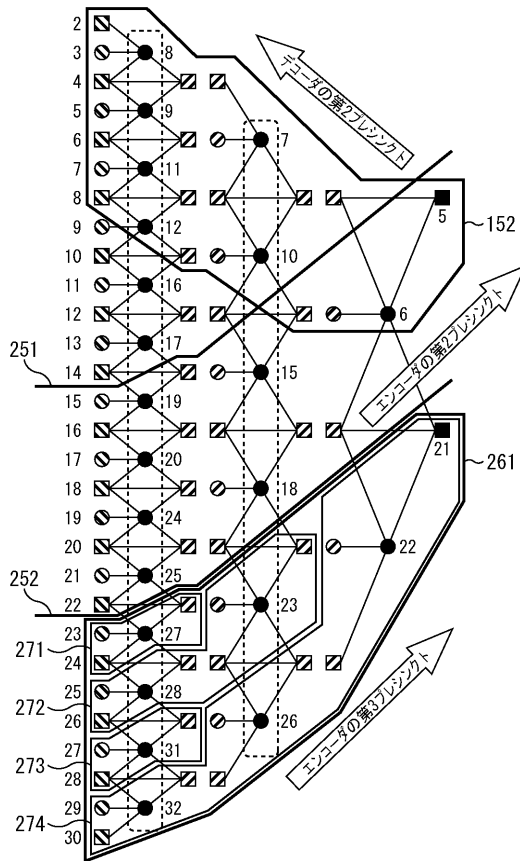
图16



【図 17】

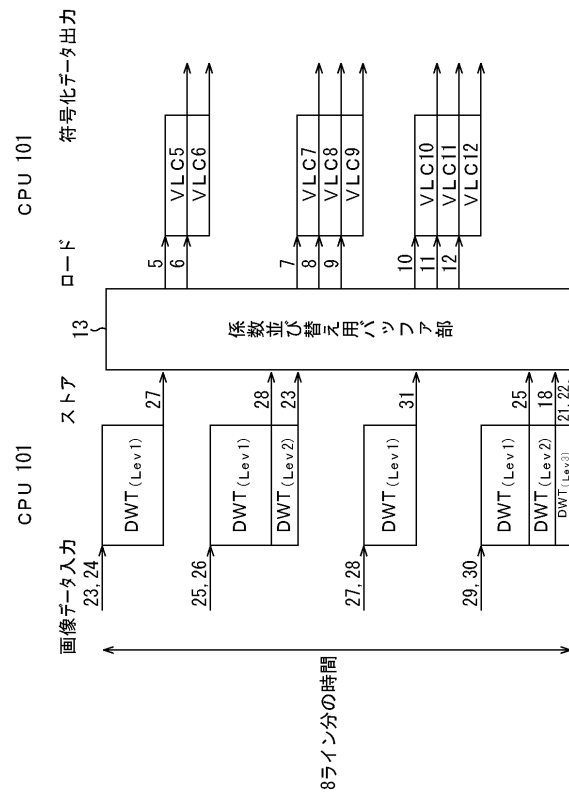
ライン番号 分割レベル1 分割レベル2 分割レベル3

図17



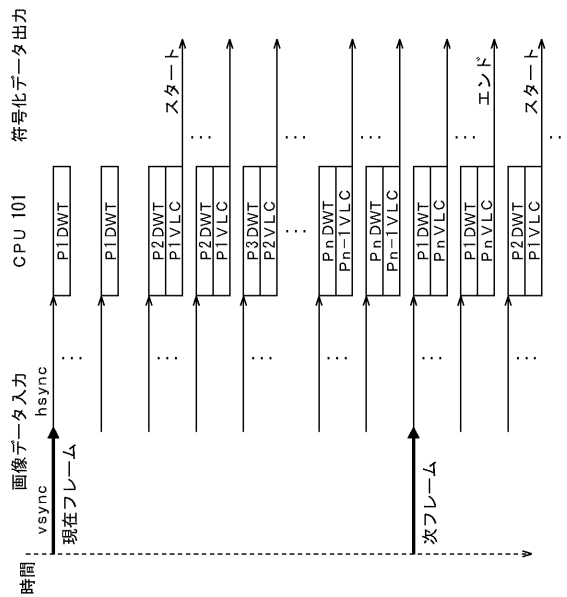
【図 18】

図18



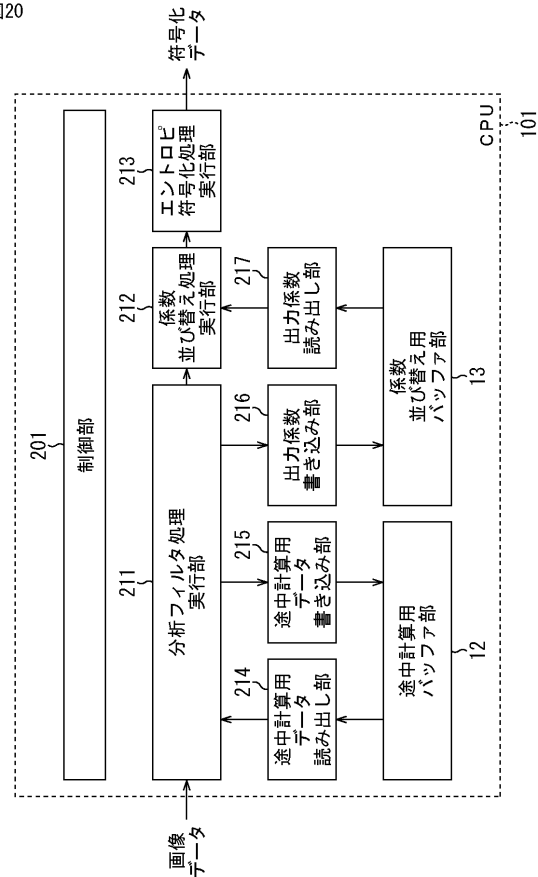
【図 19】

図19



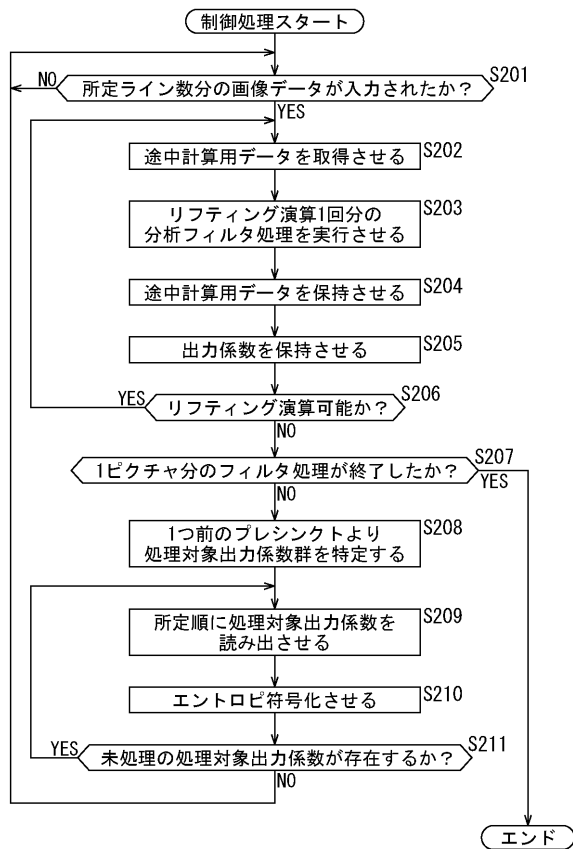
【図 20】

図20



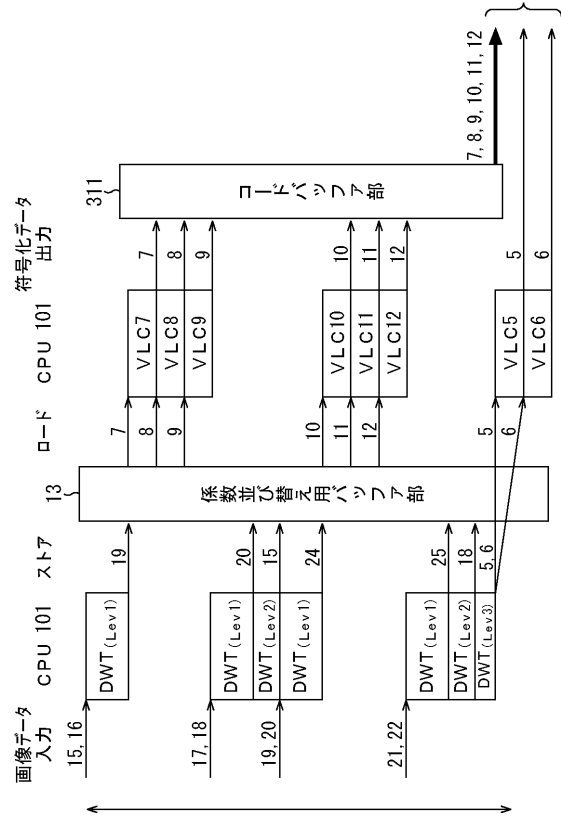
【図 2 1】

図21



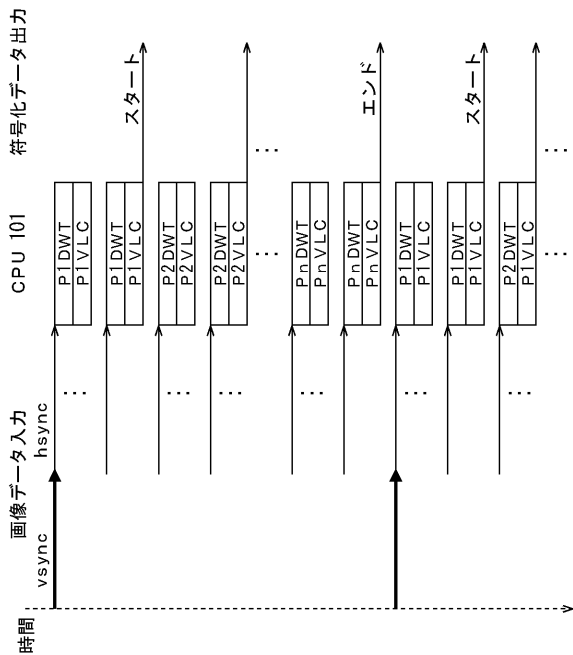
【図 2 2】

図22



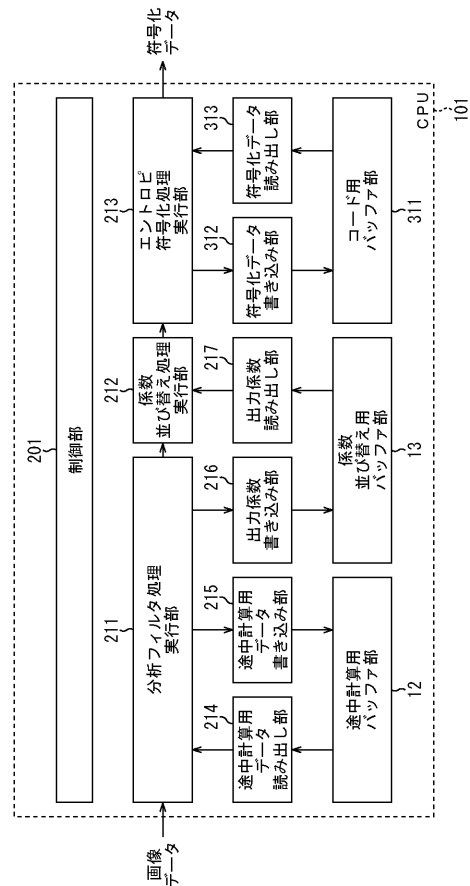
【図 2 3】

図23



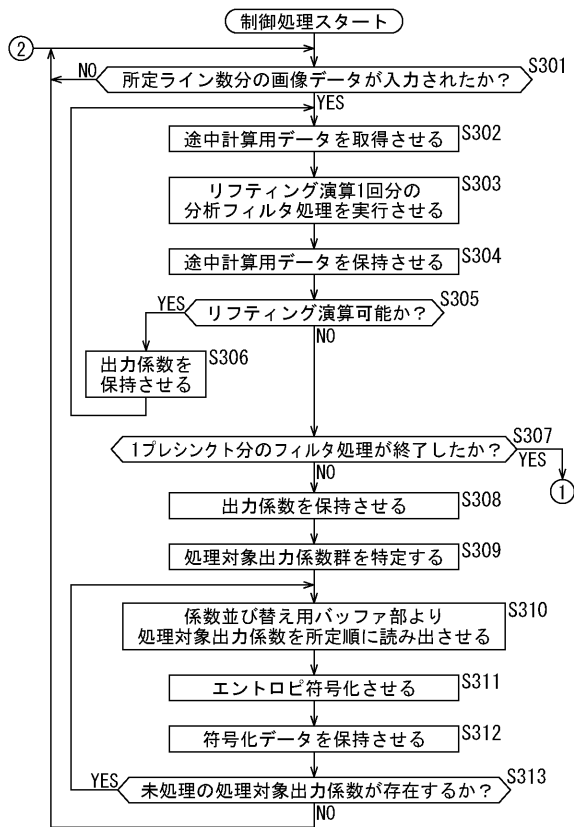
【図 2 4】

図24



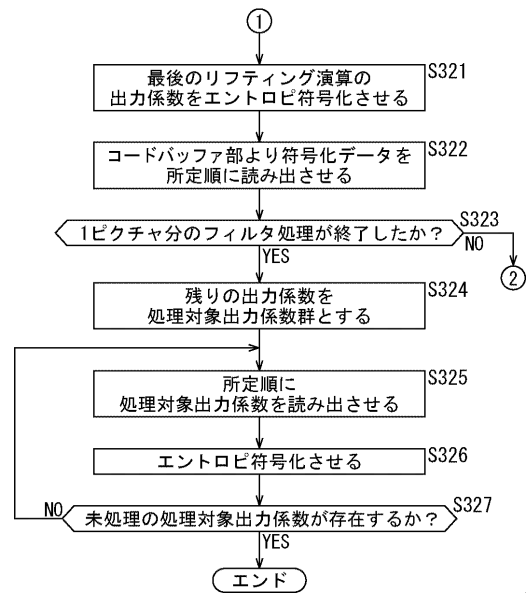
【図 25】

図25



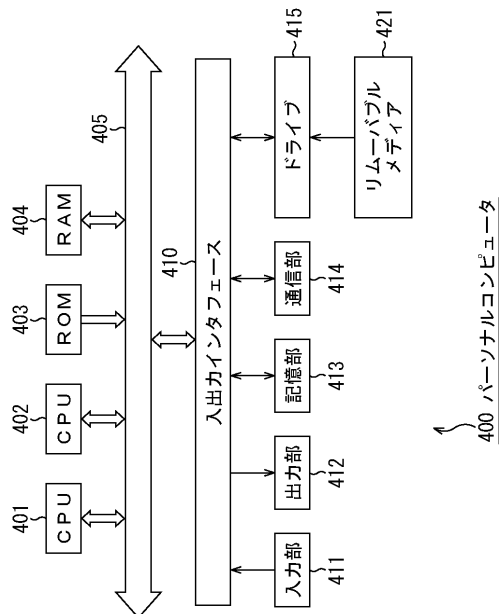
【図 26】

図26



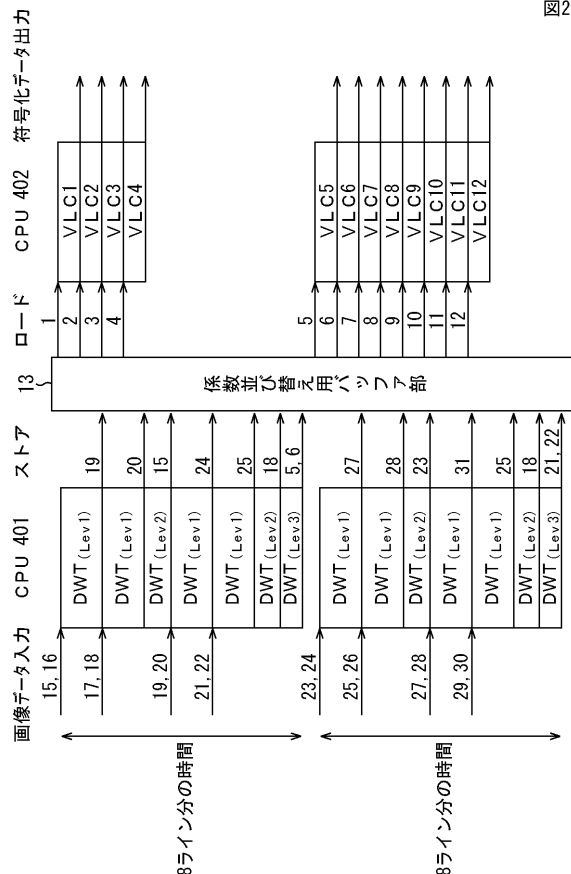
【図 27】

図27



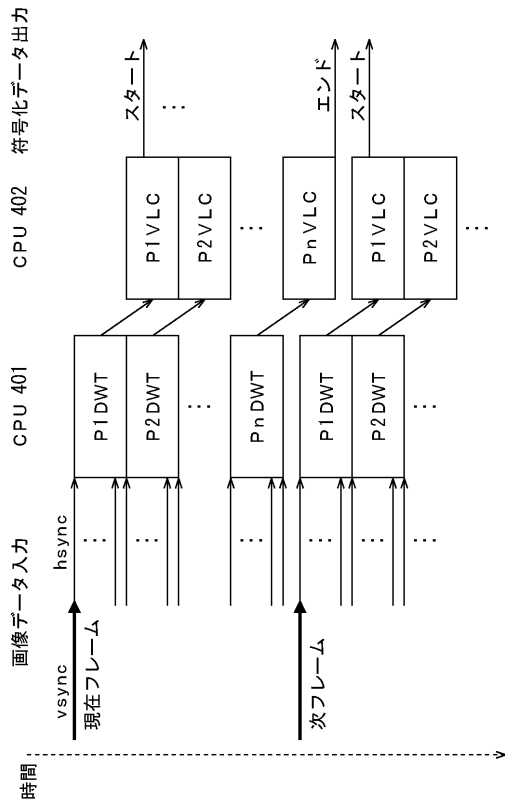
【図 28】

図28



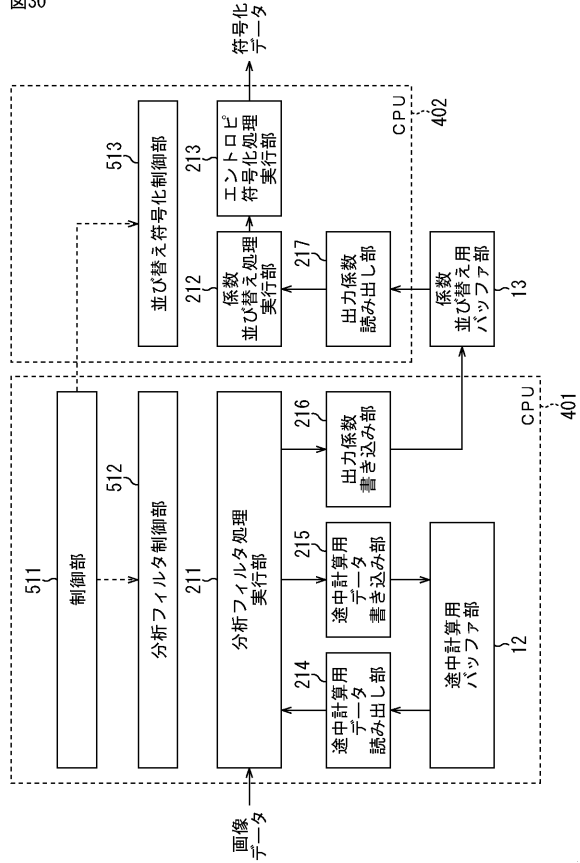
【図 29】

図29



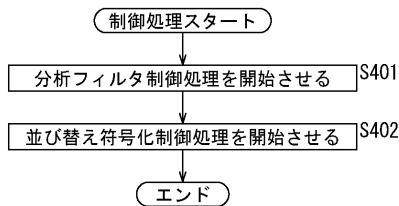
【図 30】

図30



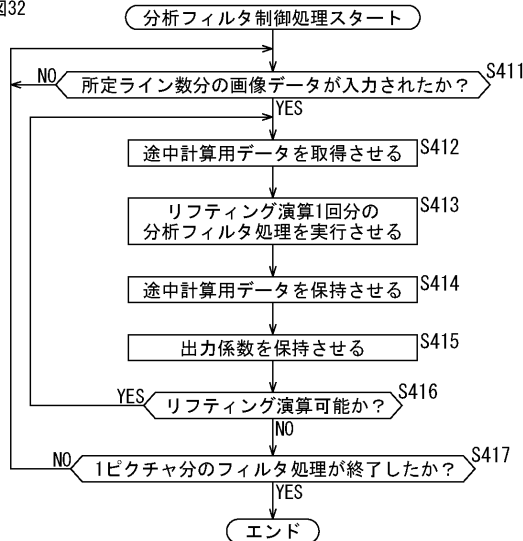
【図 31】

図31



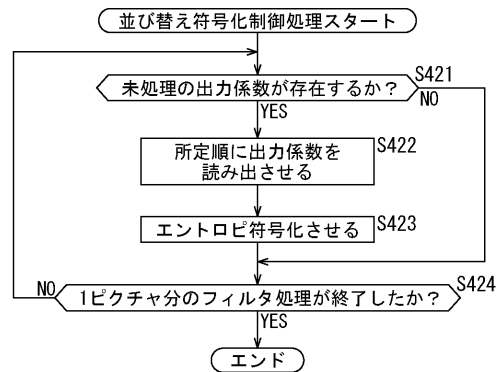
【図 32】

図32



【図 33】

図33



フロントページの続き

(56)参考文献 特開2001-326936(JP,A)
特開平10-063643(JP,A)
特開2005-217872(JP,A)
特開2007-049485(JP,A)
特開2003-274185(JP,A)
特開2000-115770(JP,A)
特開2000-299863(JP,A)
特開2001-218208(JP,A)
特開2003-319185(JP,A)
特開2004-242287(JP,A)
特開2004-040388(JP,A)
特開2004-194224(JP,A)

(58)調査した分野(Int.Cl., DB名)

H04N7/24-7/68
H04N1/41-1/419
H03M3/00-11/00