(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第6681383号
(P6681383)
(45)発行日　令和2年4月15日(2020.4.15)　(24)登録日　令和2年3月25日(2020.3.25)

(51)Int.Cl.		F I	
H04N 19/50	(2014.01)	H04N	19/50
H04N 19/46	(2014.01)	H04N	19/46
H04N 19/103	(2014.01)	H04N	19/103
H04N 19/146	(2014.01)	H04N	19/146
H04N 19/176	(2014.01)	H04N	19/176

請求項の数 30　(全 52 頁)　最終頁に続く

(21)出願番号　特願2017-501351 (P2017-501351)
(86)(22)出願日　平成27年7月21日 (2015.7.21)
(65)公表番号　特表2017-526252 (P2017-526252A)
(43)公表日　平成29年9月7日 (2017.9.7)
(86)国際出願番号　PCT/EP2015/025053
(87)国際公開番号　WO2016/012105
(87)国際公開日　平成28年1月28日 (2016.1.28)
審査請求日　平成30年5月10日 (2018.5.10)
(31)優先権主張番号　1412937.3
(32)優先日　平成26年7月21日 (2014.7.21)
(33)優先権主張国・地域又は機関
英国 (GB)

(73)特許権者　513156386
グルロジック　マイクロシステムズ　オーワイ
Ｇｕｒｕｌｏｇｉｃ　Ｍｉｃｒｏｓｙｓｔｅｍｓ　Ｏｙ
フィンランド共和国　２０１００　トゥルク　リンナンカツ　３４
Ｌｉｎｎａｎｋａｔｕ　３４　２０１００　Ｔｕｒｋｕ　ＦＩＮＬＡＮＤ
(74)代理人　100127188
弁理士　川守田　光紀
(72)発明者　カレヴォ　オッシ
フィンランド共和国　ＦＩＮ－３７８００　アカー　ケトゥンハンタ　１

最終頁に続く

(54)【発明の名称】エンコーダ、デコーダ、および方法

(57)【特許請求の範囲】

【請求項１】

入力データを符号化して対応する符号化データを生成するエンコーダ（１００）であって、前記入力データを処理し、少なくとも１つのDelta符号化アルゴリズムを用いて、前記入力データの少なくとも一部の元の値を、元の値の値範囲に比して拡大していない値範囲で表現されたデルタ値へと符号化し、前記入力データの後続の１つ以上の部分の符号化に用いる１つ以上の予測子を生成するように動作し、さらに、少なくとも１つのエントロピー符号化アルゴリズムを利用することによって、前記少なくとも１つのDelta符号化アルゴリズムおよび前記１つ以上の予測子によって生成されたデータを符号化して、前記符号化データを生成するように動作し、

前記少なくとも１つのDelta符号化アルゴリズムは、QDelta，DDelta，IDelta，PDeltaのいずれかの型のDelta符号化アルゴリズムであり、

前記少なくとも１つのDelta符号化アルゴリズムの選択肢を示す情報が前記符号化データに含まれ、

前記１つ以上の予測子は、

（ｉ）　１つ以上の時間的予測子、

（ｉｉ）　量子化の対象となる、１つ以上のローカル空間的予測子、

（ｉｉｉ）事前計算された値を用いる、１つ以上のローカル空間的予測子、

のうち少なくとも１つを含む、

ことを特徴とするエンコーダ（１００）。

【請求項２】
前記ODelta型のDelta符号化アルゴリズムは、
（ａ）　データ処理構成を用いて、前記入力データに、任意の形式の差分符号化および／または合算符号化を適用し、１つ以上の対応する符号化列を生成することと、
（ｂ）　前記データ処理構成を用いて、前記１つ以上の対応する符号化列に最大値のラップアラウンドおよび／または最小値のラップアラウンドを適用することと、
を含む、請求項１に記載のエンコーダ（１００）。
【請求項３】
前記DDelta型のDelta符号化アルゴリズムは、Delta値が負の値しか含まない場合にDelta値を符号なしで表現することを含み、
前記IDelta型のDelta符号化アルゴリズムは、Delta値が正の値しか含まない場合にDelta値を符号なしで表現することを含み、
前記PDelta型のDelta符号化アルゴリズムは、元の値の値範囲内でDelta値を表現するためにオフセットを使用することを含む、
請求項１または２に記載のエンコーダ（１００）。
【請求項４】
前記入力データを符号化して前記符号化データを生成する際に、少なくとも１つの量子化アルゴリズムを利用するように動作し、前記少なくとも１つの量子化アルゴリズムによって、前記入力データの非可逆的符号化を実行することを特徴とする、請求項１から３のいずれかに記載のエンコーダ（１００）。
【請求項５】
前記入力データに存在する、異なるデータ構造を持つデータを符号化するために、異なるアルゴリズムを利用するように動作することを特徴とする、請求項１から４のいずれかに記載のエンコーダ（１００）。
【請求項６】
前記入力データをブロック単位で符号化する際に、レート歪み（rate distortion：ＲＤ）最適化を利用するように動作することを特徴とする、請求項１から５のいずれかに記載のエンコーダ（１００）。
【請求項７】
前記ＲＤ最適化は前記エンコーダ（１００）内で計算され、以下の式の値Ｖを最小化し、
$V = D + \lambda * R$
ここで歪み（Ｄ）は、前記入力データと、前記符号化データに符号化されて復号データに復号された前記入力データの表現との二乗誤差の和であり、量（Ｒ）は、ビットとして測定される符号化データの量であることを特徴とする、請求項６に記載のエンコーダ（１００）。
【請求項８】
前記少なくとも１つのDelta符号化アルゴリズムおよび／または前記少なくとも１つのエントロピー符号化アルゴリズムは、ＤＣ法、スライド法、マルチレベル法、離散コサイン変換（discrete cosine transform：ＤＣＴ）法、ライン法、スケール法、データベース法、レンジ符号化、ハフマン符号化、連長（run-length encoding：ＲＬＥ）符号化、順次連長（serial run-length encoding：ＳＲＬＥ）符号化のうち少なくとも１つを利用するように動作することを特徴とする、請求項１から７のいずれかに記載のエンコーダ（１００）。
【請求項９】
ＹＵＶチャネルとＢＧＲチャネルとの少なくとも１つに対応するデータ構造を含む前記入力データを符号化するように動作することを特徴とする、請求項１から８のいずれかに記載のエンコーダ（１００）。
【請求項１０】
前記チャネルのデータをＹ、Ｕ、Ｖ、またはＧ、Ｂ、Ｒの順序で符号化するように動作

することを特徴とする、請求項９に記載のエンコーダ（１００）。
【請求項１１】
エンコーダ（１００）を用いて入力データを符号化し、対応する符号化データを生成する方法であって、
（ｉ）　前記エンコーダ（１００）を用いて前記入力データを処理し、前記入力データの少なくとも一部の元の値を、少なくとも１つのDelta符号化アルゴリズムを用いて、元の値の値範囲に比して拡大していない値範囲で表現されたデルタ値へと符号化することと、
（ｉｉ）　前記エンコーダ（１００）を用いて、前記入力データの後続の１つ以上の部分の符号化に用いる１つ以上の予測子を生成することと、
（ｉｉｉ）前記エンコーダ（１００）を用いて、少なくとも１つのエントロピー符号化アルゴリズムを利用することによって、前記少なくとも１つのDelta符号化アルゴリズムおよび前記１つ以上の予測子によって生成されたデータを符号化して、前記符号化データを生成することとを含み、
前記少なくとも１つのDelta符号化アルゴリズムは、ODelta，DDelta，IDelta，PDeltaのいずれかの型のDelta符号化アルゴリズムであり、
前記少なくとも１つのDelta符号化アルゴリズムの選択肢を示す情報が前記符号化データに含まれ、
前記１つ以上の予測子は、
（ｉ）　１つ以上の時間的予測子、
（ｉｉ）　量子化の対象となる、１つ以上のローカル空間的予測子、
（ｉｉｉ）事前計算された値を用いる、１つ以上のローカル空間的予測子、
のうち少なくとも１つを含む、
ことを特徴とする方法。
【請求項１２】
前記ODelta型のDelta符号化アルゴリズムは、
（ａ）　データ処理構成を用いて、前記入力データに、任意の形式の差分符号化および／または合算符号化を適用し、１つ以上の対応する符号化列を生成することと、
（ｂ）　前記データ処理構成を用いて、前記１つ以上の対応する符号化列に最大値のラップアラウンドおよび／または最小値のラップアラウンドを適用することと、
を含む、請求項１１に記載の方法。
【請求項１３】
前記DDelta型のDelta符号化アルゴリズムは、Delta値が負の値しか含まない場合にDelta値を符号なしで表現することを含み、
前記IDelta型のDelta符号化アルゴリズムは、Delta値が正の値しか含まない場合にDelta値を符号なしで表現することを含み、
前記PDelta型のDelta符号化アルゴリズムは、元の値の値範囲内でDelta値を表現するためにオフセットを使用することを含む、
請求項１１に記載の方法。
【請求項１４】
前記入力データを符号化して前記符号化データを生成する際に、前記エンコーダ（１００）に少なくとも１つの量子化アルゴリズムを利用させ、前記少なくとも１つの量子化アルゴリズムによって、前記入力データの非可逆的符号化を前記エンコーダ（１００）に実行させる構成を含むことを特徴とする、請求項１１から１３のいずれかに記載の方法。
【請求項１５】
前記入力データに存在する、異なるデータ構造を持つデータを符号化するために、前記エンコーダ（１００）に異なるアルゴリズムを利用させる構成を含むことを特徴とする、請求項１１から１４のいずれかに記載の方法。
【請求項１６】
前記入力データをブロック単位で符号化する際に、前記エンコーダ（１００）にレート

歪み（rate distortion：ＲＤ）最適化を利用させる構成を含むことを特徴とする、請求項１１から１５のいずれかに記載の方法。

【請求項１７】

前記ＲＤ最適化は前記エンコーダ（１００）内で計算され、以下の式の値Ｖを最小化し、

Ｖ＝Ｄ＋λ＊Ｒ

ここで歪み（Ｄ）は、前記入力データと、前記符号化データに符号化されて復号データに復号された前記入力データの表現との二乗誤差の和であり、量（Ｒ）は、ビットとして測定される符号化データの量である、

ことを特徴とする、請求項１６に記載の方法。

【請求項１８】

前記少なくとも１つのDelta符号化アルゴリズムおよび／または前記少なくとも１つのエントロピー符号化アルゴリズムは、ＤＣ法、スライド法、マルチレベル法、離散コサイン変換（discrete cosine transform：ＤＣＴ）法、ライン法、スケール法、データベース法、レンジ符号化、ハフマン符号化、連長（run-length encoding：ＲＬＥ）符号化、順次連長（serial run-length encoding：ＳＲＬＥ）符号化のうち少なくとも１つを利用するように動作することを特徴とする、請求項１１から１７のいずれかに記載の方法。

【請求項１９】

前記エンコーダ（１００）に、ＹＵＶチャネルとＢＧＲチャネルとの少なくとも１つに対応するデータ構造を含む前記入力データを符号化させる構成を含むことを特徴とする、請求項１１から１８のいずれかに記載の方法。

【請求項２０】

前記エンコーダ（１００）に、前記チャネルのデータをＹ、Ｕ、Ｖ、またはＧ、Ｂ、Ｒの順序で符号化させる構成を含むことを特徴とする、請求項１９に記載の方法。

【請求項２１】

符号化データを復号して対応する復号データを生成するデコーダ（１２０）であって、少なくとも１つのエントロピー復号アルゴリズムを前記符号化データに適用して処理して、処理済みデータを生成し、少なくとも１つのDelta復号アルゴリズムと共に１つ以上の予測子を用いて、前記処理済みデータを復号して、前記復号データを生成するように動作し、前記処理されたデータは、前記符号化データが生成される元のデータの値範囲に比して拡大していない値範囲で表現されたデルタ値を含み、

前記少なくとも１つのDelta復号アルゴリズムは、逆ODelta，逆DDelta，逆IDelta，逆PDeltaのいずれかの型のDelta復号アルゴリズムであり、

前記少なくとも１つのDelta復号アルゴリズムの選択肢を示す情報が前記符号化データに含まれ、

前記１つ以上の予測子は、

（ｉ）　１つ以上の時間的予測子、

（ｉｉ）　量子化の対象となる、１つ以上のローカル空間的予測子、

（ｉｉｉ）事前計算された値を用いる、１つ以上のローカル空間的予測子、

のうち少なくとも１つを含む、

ことを特徴とするデコーダ（１２０）。

【請求項２２】

前記逆ODelta型のDelta復号アルゴリズムは、

（ａ）　データ処理構成を用いて、前記符号化データに、任意の形式の差分復号および／または合算復号を適用し、１つ以上の対応する復号列を生成することと、

（ｂ）　前記データ処理構成を用いて、前記１つ以上の対応する復号列に最大値のラップアラウンドおよび／または最小値のラップアラウンドを適用することと、を含む、請求項２１に記載のデコーダ（１２０）。

【請求項２３】

前記逆DDelta型のDelta復号アルゴリズムは、Delta値が負の値しか含まない場合にDelt

a値を復号することを含み、
前記逆IDelta型のDelta復号アルゴリズムは、Delta値が正の値しか含まない場合にDelt
a値を復号することを含み、
前記逆PDelta型のDelta復号アルゴリズムは、前記元の値の値範囲内で表現されたDelta
値を復号するためにオフセットを使用することを含む、
請求項２１または２２に記載のデコーダ（１２０）。
【請求項２４】
前記符号化データを復号して前記復号データを生成する際に、少なくとも１つの量子化アルゴリズムを利用するように動作し、前記少なくとも１つの量子化アルゴリズムによって、前記符号化データの非可逆的復号を実行することを特徴とする、請求項２１から２３のいずれかに記載のデコーダ（１２０）。
【請求項２５】
前記符号化データに存在する、異なるデータ構造を持つデータを復号するために、異なるアルゴリズムを利用するように動作することを特徴とする、請求項２１から２４のいずれかに記載のデコーダ（１２０）。
【請求項２６】
前記少なくとも１つのDelta復号アルゴリズムおよび／または前記少なくとも１つのエントロピー復号アルゴリズムは、ＤＣ法、スライド法、マルチレベル法、離散コサイン変換（discrete cosine transform：ＤＣＴ）法、ライン法、スケール法、データベース法、レンジ符号化、ハフマン符号化、連長（run-length encoding：ＲＬＥ）符号化、順次連長（serial run-length encoding：ＳＲＬＥ）符号化のうち少なくとも１つを利用するように動作することを特徴とする、請求項２１から２５のいずれかに記載のデコーダ（１２０）。
【請求項２７】
ＹＵＶチャネルとＢＧＲチャネルとの少なくとも１つに対応するデータ構造を含む前記符号化データを復号するように動作することを特徴とする、請求項２１から２６のいずれかに記載のデコーダ（１２０）。
【請求項２８】
前記チャネルのデータをＹ、Ｕ、Ｖ、またはＧ、Ｂ、Ｒの順序で復号するように動作することを特徴とする、請求項２７に記載のデコーダ（１２０）。
【請求項２９】
デコーダ（１２０）において符号化データを復号して対応するデータを生成する方法であって、少なくとも１つのエントロピー復号アルゴリズムを前記符号化データに適用して処理して、処理済みデータを生成し、少なくとも１つのDelta復号アルゴリズムと共に１つ以上の予測子を用いて、前記処理済みデータを復号して、復号データを生成することを含み、前記処理されたデータは、前記符号化データが生成される元のデータの値範囲に比して拡大していない値範囲で表現されたデルタ値を含み、
前記少なくとも１つのDelta復号アルゴリズムは、逆QDelta，逆DDelta，逆IDelta，逆PDeltaのいずれかの型のDelta復号アルゴリズムであり、
前記少なくとも１つのDelta復号アルゴリズムの選択肢を示す情報が前記符号化データに含まれ、
前記１つ以上の予測子は、
（ｉ）　１つ以上の時間的予測子、
（ｉｉ）　量子化の対象となる、１つ以上のローカル空間的予測子、
（ｉｉｉ）事前計算された値を用いる、１つ以上のローカル空間的予測子、
のうち少なくとも１つを含む、
ことを特徴とする方法。
【請求項３０】
装置の処理手段に実行されると、前記装置に、請求項１１～２０，２９のいずれか１項に記載の方法を遂行させるように構成されるプログラム命令を備える、コンピュータプロ

グラム。

【発明の詳細な説明】

【技術分野】

【０００１】

本願に開示される事項は、データを符号化する方法、例えば、１つ以上の予測子を利用するDelta符号化を用いてデータを符号化する方法に関する。また本願に開示される事項は、データを復号する方法、例えば、１つ以上の予測子を利用するDelta符号化を用いてデータを復号する方法に関する。さらに本願に開示される事項は、前述の方法を実行するシステム、装置、および機器に関する。さらに本願に開示される事項は、前述の方法を実行するために、処理ハードウェアを備えるコンピュータ装置によって実行可能なコンピュータ可読命令が格納された非一時的コンピュータ可読格納媒体を含む、コンピュータプログラム製品にも関する。

【背景】

【０００２】

一般に、ＭＰＥＧ－４、Ｈ．２６４、ＶＣ－１、ＨＥＶＣ、ＶＰ９などの従来のビデオコーデックは、画像ブロックの動き推定に前のフレームを利用できる。これらのコーデック例の名称には登録商標が含まれている。動き推定および動き補償は、各動画フレームのブロック単位で実行される。同様に、任意の現在のデータブロックやデータパケットを符号化する際に、重複排除方法やデータベースを用いた処理を用いて、符号化処理済みのデータブロックやデータパケットを利用することもできる。動画や類似の種類のコンテンツに含まれるデータ記号のエントロピーを低減するために、Delta符号化を用いることができる。以下に詳細に説明するとおり、データ記号のエントロピーをさらに低減するために、ODelta符号化を用いてもよい。ODelta符号化によって、個々のビットのエントロピーを低減することもできる。ODelta符号化に関連する方法の詳細は、付属書類１において後述する。差分パルス符号変調（differential pulse-code modulation：ＤＰＣＭ）式の方法、すなわちDelta符号化およびODelta符号化では、エントロピーエンコーダ用の符号化データ値を作成する際に前のデータ値を用いる。

【０００３】

近年における急速なデータ量の増大および転送の高速化のため、データ圧縮の必要性も増大し、データ圧縮の効率を向上するために改良された新たな方法が必要とされている。データは、例えば、１つ以上のセンサから取得されてもよく、画像、動画、音声、測定データ、または様々な種類のバイナリデータ、ＡＳＣＩＩデータなどであってもよい。取得されたセンサデータと抽象データの組合せであってもよい。

【０００４】

現在、データの符号化に利用できる符号化処理方法は複数あるが、異なる種類のデータすべてに十分な圧縮率を提供する方法はない。任意の現在のチャネル、フレーム、データブロック、またはデータパケットを符号化する場合、異なるデータチャネルまたはフレーム、例えば、色、画像チャネル、音声チャネル、並列データ測定、動画内の個々の画像、音声内の個々のパケット、３次元（three-dimensional：３Ｄ）画像、３Ｄ音声などを別々に符号化処理し、同時に、符号化処理済みのチャネル、フレーム、データブロック、またはデータパケットの情報も用いる必要があることが多い。既知のデータ符号化方法は、非常に多様なデータ構造を備える入力データに対処できるほど十分に汎用的ではない。

【０００５】

同様に、ＤＰＣＭ（http://en.wikipedia.org/wiki/DPCM）などの既知の方法を用いるより、符号化済みの空間情報を符号化に用いるほうが有利であり、より効率的である。また、データの可逆的および非可逆的な符号化を可能にする、単純かつ効率的な符号化方法および復号方法が求められている。付属書類１に詳述するように、ODelta法による方法があるが、例えば、典型的な既知のDelta符号化を修正し、ＤＰＣＭのDelta符号化を様々な予測子と共に用いることができるようにして、ODelta法をより効率的に用いる必要がある。そのような手法により、例えばＤＰＣＭのような従来の既知のDelta符号化よりも、記

号間の負および正の差分値または合算値に対してより効果的なエントロピー低減が可能になる。

【０００６】

既知の方法には、高度な予測方法を利用したデータの再利用と、量子化ありまたはなしの残差データの効率的なエントロピー低減、という２つの特性を組み合わせることができるものが存在しない。この組合せは、方法の選択肢、すなわちフレーム、チャネル、データブロック、またはデータパケット用の方法と、符号化した残差値、すなわち、動きベクトル、選択記号、データベース参照などを含まない値のみを提供または格納することによって実現される。次のいずれかの理由から、残差符号化も不要になる場合がある。

（ｉ）　予測子が最適である。

（ｉｉ）　残差がすべてのデータ値において一定であり、１つの値だけで提供できる。

（ｉｉｉ）量子化ありまたはなしの残差が、品質パラメータに基づく誤差閾値を下回る。

【摘要】

【０００７】

本発明は、Delta符号化アルゴリズムの使用に基づいて、データを符号化する改良された方法を提供することを目的とする。

【０００８】

また、本発明は、Delta符号化アルゴリズムの使用に基づいて、データを符号化する改良されたエンコーダを提供することを目的とする。

【０００９】

さらに、本発明は、逆Delta符号化アルゴリズムの使用に基づいて、データを復号する改良された方法を提供することを目的とする。

【００１０】

さらに、本発明は、逆Delta符号化アルゴリズムの使用に基づいて、データを復号する改良されたデコーダを提供することを目的とする。

【００１１】

第１の態様によると、入力データ（Ｄ１）を符号化して対応する符号化データ（Ｅ２）を生成するエンコーダ（１００）であって、前記入力データ（Ｄ１）を処理し、少なくとも１つのDelta符号化アルゴリズムを用いて前記入力データ（Ｄ１）の少なくとも一部を符号化し、前記入力データ（Ｄ１）の後続の１つ以上の部分の符号化に用いる１つ以上の予測子を生成するように動作し、さらに、少なくとも１つのエントロピー符号化アルゴリズムを利用することによって、前記少なくとも１つのDelta符号化アルゴリズムおよび前記１つ以上の予測子によって生成されたデータを符号化して、前記符号化データ（Ｅ２）を生成するように動作し、前記１つ以上の予測子は、

（ｉ）　１つ以上の時間的予測子、

（ｉｉ）　複数の値を用いる、１つ以上のローカル空間的予測子、

（ｉｉｉ）量子化の対象となる、１つ以上のローカル空間的予測子、

（ｉｖ）　事前計算された値を用いる、１つ以上のローカル空間的予測子、

のうち少なくとも１つを含むことを特徴とするエンコーダ（１００）が提供される。

【００１２】

本発明は、Delta符号化と、１つ以上のローカル空間的予測子の生成と、デルタ（delta）値の多様な提供方法の利用と、エントロピー符号化との組合せによって、効率の高いデータ符号化を提供できるという点で有利である。

【００１３】

「ローカル空間的予測子」は、短縮形として「ローカル予測子」または「空間的予測子」と呼んでもよい。

【００１４】

事前計算された値について、これらの値は、前述のデータ処理を実行する前に空間的位置に基づいて決定できる任意の値であってよい。すなわち、これらの値は、付属書類１に記載するように、0Delta符号化におけるようなローカル空間的予測子ではない。このよう

な事前計算された値には、例えば、符号化する現在のデータブロックに対して左側または上にある前のブロック、任意の前のチャネル、ビュー、またはフレーム内にある同じデータブロックの位置、任意のデータフレーム内の内部動き予測などがある。

【００１５】

前記方法による予測は、通常、事前に一度だけで決定される。すなわち、任意のブロックに対する予測は、時間的予測または空間的予測によって事前に一度だけで決定される。前述のブロック、チャネル、ビュー、およびフレームに加えて、動き推定も便宜上、時間的予測とみなされる。動き推定は空間的に実行することもできるが、その場合、前述の項目（ｉｖ）に定義する空間的予測子に関するものとなる。つまり、動き推定予測は、必ずしも時間的に相互に異なる必要があるわけではない。この空間的予測は、付属書類１に記述する、例えばODelta法において用いる「ローカル空間的予測」とは異なる。「ローカル空間的予測」を用いる場合、方法の処理中に予測が行われ、前の値が処理されるまで予測子の値を利用できない。これは時間的予測および他の空間的予測と比べて異なる点である。ODeltaと同様に（付属書類１を参照）、予測を処理の実行中に決定してもよい。その場合、Delta符号化におけるローカル空間的予測子が用いられる。

【００１６】

前記エンコーダにおいて、数値の列を含む入力データ（ＤＡ１）を符号化して対応する符号化出力データ（ＤＡ２またはＤＡ３）を生成する前記少なくとも１つのDelta符号化アルゴリズムは、ODelta符号化アルゴリズムとして実装され、前記ODelta符号化アルゴリズムは、

（ａ）　データ処理構成を用いて、前記入力データ（ＤＡ１）に、任意の形式の差分符号化および／または合算符号化を適用し、１つ以上の対応する符号化列を生成することと、

（ｂ）　前記データ処理構成を用いて、前記１つ以上の対応する符号化列に最大値のラップアラウンドおよび／または最小値のラップアラウンドを適用し、前記符号化出力データ（ＤＡ２またはＤＡ３）を生成することと、

を含むことを特徴としてもよい。

【００１７】

前記Delta符号化アルゴリズムは、以下のように用いてもよい。量子化の適用の有無にかかわらず、任意の元の値と、対応する予測値との間の差分が常にゼロか正である場合、または、任意の元の値と、対応する予測値との間の差分がゼロか負である場合、符号化処理アルゴリズム情報と共に、デルタ値の符号、例えば符号ビットのみを表現および提供することができる。そのような場合は、ラッピングなしでも値が常に任意のビット数範囲内に収まるため、ODeltaアルゴリズムにおいて利用されるようなラッピングは完全に不要である。このように用いられるアルゴリズムは、便宜上、例えばIDelta（incremental delta：増分Delta）法およびDDelta（decremental delta：減分Delta）法と呼ばれる。IDelta法は正のデルタ値のみを提供し、DDelta法は負のデルタ値のみ提供するが、しばしば、符号、すなわち符号ビットを有利に交換することがある。この交換はDDelta値のみに対して行われ、かつDDelta値を提供するときにのみ行われる。前述のいずれの方法も、量子化の有無にかかわらず、ゼロ記号／値と共に不変の値を常に提供できる。すなわち、このような場合、これらのデルタ値間の差分は用いられる量子化以下である。

【００１８】

また、デルタ値を基礎（pedestal）値と共に用いてもよい。つまり、負のデルタ値と正のデルタ値の両方が生じ、それらの絶対値が、符号化処理されているデータのダイナミックレンジとビット深度に関して小さい場合、量子化の有無にかかわらず、変化の基礎値、すなわち最大の負の変化を提供すると有利であることがある。その後、前述のIDelta法と同様に、正の変化の値のみを提供することができる。基礎値を利用するこのような方法を、便宜上、「PDelta法」と呼ぶことができる。PDelta法を用いる場合、元のODeltaアルゴリズム（付属書類１を参照）を用いる場合よりも、データのダイナミックレンジ／ビット深度を低減できることが多い。つまり、そのデータの最大値が小さくなり、従って可能な最小値と可能な最大値の間の差分も同様に小さくなるため、より少ないビットでデータを

表現できる。元のODelta法（付属書類１を参照）は、生じる値のダイナミックレンジ、すなわち提供するODelta値の値範囲を常に表現および提供しなければならないため、このような利点が得られる。

【００１９】

前述のIDelta、DDelta、およびPDelta法は、用いる予測が、例えば、前のブロック、チャネル、画像、または、これらの方法を用いる前に決定され宣言される他の一式の値である場合に、本願の実施例に用いると有利である。したがって、符号化処理されている値と予測値との間の差分は、一度だけで容易に決定され、その値または差分の予測は、符号化処理されている他の値にまったく依存しない。「時間的予測子」または「空間的予測子」と呼ばれるこの種の予測手段は、特に量子化との併用にも非常に適している。なぜなら、差分値の量子化は、復号される任意の独立したデータ値ひとつだけに影響し、量子化によって生じる誤差は、復号される他のデータ値に累積しないためである。量子化はローカル空間的予測子と共に用いることもできるが、その場合、量子化誤差が累積しないように、アルゴリズムにおいて次の値を予測する際に量子化誤差を考慮に入れる必要がある。

【００２０】

既知の種類のDelta符号化は、任意の現在値から予測値を引くことで実行され、これはローカル空間的予測子を用いて達成される。これらの値の差分は正の値と負の値のいずれにもなり得るため、符号を常に差分値に付して提供する必要がある。ODelta符号化（付属書類１を参照）では、現在値と予測値との差分または合算を用いることができ、また、値のラッピングを用いて値を提供できるため、値を常に正にすることができる。

【００２１】

前述のように、IDelta、DDelta、およびPDelta法は、既知のDelta法、ならびに、ラッピングを用いて予測値が提供される付属書類１に記載する前述のODelta法とは異なる。また、付属書類１に記述するODelta法では、用いられる予測子が、１つの値のみを用いるローカル空間的予測子に限られ、この予測子は常に量子化なしで用いられる。しかしながら、このような制限は本願の実施例には不要である。本願の実施例では、他の種類の予測子および／または量子化を利用し、前述のODelta法と組み合わせることもできる。また、ラッピングを利用する元のODelta法を、IDelta、DDelta、またはPDelta法において用いることも可能である。ODelta法は優れた手段であることが多く、場合によっては前述のIDelta、DDelta、およびPDelta法より優れた手段である場合もある。ラッピングを利用する元のODelta法は、１つの同じラップ値／記号によって正の差分と負の差分の両方を表現できることが多いからである。ラッピングを利用する元のODelta法は、値範囲および予測値に基づいて、正の値と負の値を後で互いに区別できるようにこれを行うが、前述のIDelta、DDelta、およびPDelta法は、ゼロ値と正の差分値、または負の差分値のみが提供されるため、それ以上の区別は不要である。

【００２２】

多くの場合、用いられる予測、または場合によっては用いられる量子化に関する情報をメソッド選択情報と共に提供してもよい。量子化に関する情報は、１つの量子化値または１つの品質値を用いることによって、データの列全体に対して一度に提供でき、有利であることが多い。そのようなメソッドの例として、IDeltaBlockFromChannel0、PDeltaChannelR_2、DDeltaFrame_4、ODeltaBlockMode、DDeltaPacketPrevQ70などがある。これらのメソッドの１つ目、すなわちIDeltaBlockFromChannel0は、任意の対応するブロック内のチャネル０の値に関する正の差分値のみを提供する。すなわち、現在符号化処理されているデータブロックの各データ値は、予測として用いられるブロックのチャネル０内の対応する位置にある値より大きいか、または少なくとも等しい。

【００２３】

これらのメソッドの２つ目、PDeltaChannelR_2は、基礎値に続いて正の差分値、すなわち現在のチャネルとＲチャネルとの差分を、さらに２で量子化したものを提供する。DDeltaFrame_4と呼ばれるメソッドは、例えば、徐々に暗くなる画像によく適している。このメソッドは、前のフレームと比較した負の値の変化を、４で量子化したものを提供する。

0DeltaBlockModeと呼ばれるメソッドは、現在のブロック領域をモード値と比較して、ラップされた差分値を提供する。このモード値を各ブロックに個別に送信／提供しても、あるいは、モード値を予測する際に、データチャネル全体のモード値またはデータフレーム全体のモード値を用いてもよい。DDeltaPacketPrevQ70は、現在のパケットを同じサイズの前のパケットと比較して負のデータ値を提供し、デルタ値の量子化は、品質係数７０によって定義される。

【００２４】

前述の例は、添付の特許請求の範囲に定義される本発明の保護の範囲を制限すると解釈されるものではない。本願の開示に従う方法の多様な実施形態を記述するために、多くの他の類似の方法を用いることができる。提供するデータを可能な限り効率的に圧縮するために、これらすべての方法と共に、通常は最大値であるデータ値制限を提供してもよい。

【００２５】

本願に開示される事項は、符号化アルゴリズム自体の動作を妨げることなく、既知の符号化方法を効率的に用いて部分的にデータを符号化する代替方法を定義し説明する。したがって、本明細書及び図面において以下に説明する実施形態の方法は、例えば、他の既知の方法と共に有利に用いられるか、または、既知の符号化処理方法に取って代わることができる。本願の方法は、様々な予測子および任意で量子化子を用いて、本願の実施例の方法の実施後に任意のエントロピーエンコーダにおいてエントロピーを確実に低減できるようにする。

【００２６】

前記エンコーダは、前記入力データ（Ｄ１）を符号化して前記符号化データ（Ｅ２）を生成する際に、少なくとも１つの量子化アルゴリズムを利用するように動作し、前記少なくとも１つの量子化アルゴリズムによって、前記入力データ（Ｄ１）の非可逆的符号化を実行してもよい。

【００２７】

前記エンコーダは、前記入力データ（Ｄ１）に存在する、相互に異なるデータ構造を持つデータを符号化するために、相互に異なるアルゴリズムを利用するように動作してもよい。

【００２８】

前記エンコーダは、前記入力データ（Ｄ１）をブロック単位で符号化する際に、レート歪み（rate distortion：ＲＤ）最適化を利用するように動作してもよい。また、前記エンコーダにおいて、前記ＲＤ最適化は前記エンコーダ内で計算され、以下の式の値Ｖを最小化してもよく、

Ｖ＝Ｄ＋λ＊Ｒ

ここで歪み（Ｄ）は、例えば、前記入力データ（Ｄ１）と、前記符号化データ（Ｅ２）に符号化されて前記復号データ（Ｄ３）に復号された前記入力データ（Ｄ１）の表現との二乗誤差（ＳＥ）の和であり、量（Ｒ）は、例えばビットとして測定される符号化データの量である。

【００２９】

前記エンコーダにおいて、前記少なくとも１つの符号化および／またはエントロピー符号化アルゴリズムは、ＤＣ法、スライド法、マルチレベル法、離散コサイン変換（discrete cosine transform：ＤＣＴ）法、ライン法、スケール法、データベース法、レンジ符号化、ハフマン符号化、連長（run-length encoding：ＲＬＥ）符号化、順次連長（serial run-length encoding：ＳＲＬＥ）符号化のうち少なくとも１つを利用するように動作してもよい。

【００３０】

前記エンコーダは、ＹＵＶチャネルとＢＧＲチャネルとの少なくとも１つに対応するデータ構造を含む前記入力データ（Ｄ１）を符号化するように動作してもよい。前記エンコーダは、前記チャネルのデータをＹ、Ｕ、Ｖ、またはＧ、Ｂ、Ｒの順序で符号化するように動作してもよい。アルファチャネル、すなわち透明チャネルを、他のチャネルとは個別

に、または一緒に符号化処理することもできる。同様に、符号化処理するデータは音声データであってもよく、その場合、例えばビット深度８、１６、または２４である音声振幅値をチャネルごとに個別に、または多数のチャネルで一度に符号化処理することもできる。したがって、符号化処理するデータが音声、画像、動画、ゲノムデータ、測定結果、テキスト、バイナリなどであるかどうかにかかわらず、符号化処理するデータのビット深度は、データ要素につき１から、例えば、２５６ビットまで様々であってもよい。

【００３１】

前記エンコーダは、前記エンコーダが、前記入力データ（Ｄ１）を符号化して前記符号化データ（Ｅ２）を生成するために利用した１つ以上の符号化アルゴリズムを示すデータを前記符号化データに含めるように動作してもよい。

【００３２】

第２の態様によると、エンコーダを用いて入力データ（Ｄ１）を符号化し、対応する符号化データ（Ｅ２）を生成する方法であって、

（ｉ）　前記エンコーダを用いて前記入力データ（Ｄ１）を処理し、前記入力データ（Ｄ１）の少なくとも一部を、少なくとも１つのDelta符号化アルゴリズムを用いて符号化することと、

（ｉｉ）　前記エンコーダを用いて、前記入力データ（Ｄ１）の後続の１つ以上の部分の符号化に用いる１つ以上の予測子を生成することと、

（ｉｉｉ）前記エンコーダを用いて、少なくとも１つのエントロピー符号化アルゴリズムを利用することによって、前記少なくとも１つのDelta符号化アルゴリズムおよび前記１つ以上の予測子によって生成されたデータを符号化して、前記符号化データ（Ｅ２）を生成することとを含み、前記１つ以上の予測子は、

（ｉ）　１つ以上の時間的予測子、

（ｉｉ）　複数の値を用いる、１つ以上のローカル空間的予測子、

（ｉｉｉ）量子化の対象となる、１つ以上のローカル空間的予測子、

（ｉｖ）　事前計算された値を用いる、１つ以上のローカル空間的予測子、

のうち少なくとも１つを含むことを特徴とする方法が提供される。

【００３３】

前記方法において、数値の列を含む入力データ（ＤＡ１）を符号化して対応する符号化出力データ（ＤＡ２またはＤＡ３）を生成する前記少なくとも１つのDelta符号化アルゴリズムは、

（ａ）　データ処理構成を用いて、前記入力データ（ＤＡ１）に、任意の形式の差分符号化および／または合算符号化を適用し、１つ以上の対応する符号化列を生成することと、

（ｂ）　前記データ処理構成を用いて、前記１つ以上の対応する符号化列に最大値のラップアラウンドおよび／または最小値のラップアラウンドを適用し、前記符号化出力データ（ＤＡ２またはＤＡ３）を生成することと、

を含むことを特徴としてもよい。

【００３４】

前記ODelta法（付属書類１を参照）は、以下のように用いてもよい。量子化の利用の有無にかかわらず、任意の元の値と、対応する予測値との間の差分が常にゼロか正である場合、または、前記元の値と前記予測値との間の差分がゼロか負である場合、符号化方法情報と共に、デルタ値の符号、すなわち符号ビットのみを表現および提供することができる。そのような場合は、ラッピングなしでも値が常に任意のビット数範囲内に収まるため、ラッピングは完全に不要である。

【００３５】

このように用いられる方法は、便宜上、例えばIDelta（incremental delta：増分Delta）法およびDDelta（decremental delta：減分Delta）法と呼ぶことができる。IDelta法は正のデルタ値のみを提供し、DDelta法は負のデルタ値のみ提供するが、しばしば、符号、すなわち符号ビットを有利に交換することがある。前述のいずれの方法も、量子化の有無にかかわらず、ゼロ記号／値と共に不変の値を常に提供できる。すなわち、このような場

合、これらの値間の差分は用いられる量子化以下である。

【００３６】

また、デルタ値を基礎値と共に用いてもよい。つまり、負のデルタ値と正のデルタ値の両方が生じ、それらの絶対値が、符号化処理されているデータのダイナミックレンジとビット深度に関して小さい場合、量子化の有無にかかわらず、変更の基礎値、すなわち最大の負の変化を提供すると有利であることがある。その後、前述のIDelta法と同様に、正の変化の値のみを提供することができる。そのような基礎法は、例えば「PDelta法」と呼ぶことができる。PDelta法を用いる場合、付属書類１に記述するような元のODelta法を用いる場合よりも、データのダイナミックレンジ／ビット深度を低減できることが多い。つまり、データの最大値がより小さくなり、可能な最小値と最大値との差分が小さくなるため、より少ないビットによってデータを表現できる。上記のようになるのは、ODelta法は常に、生じる値のダイナミックレンジ、すなわち提供されるODelta値の値範囲を表現および提供する必要があるためである。

【００３７】

前述のIDelta、DDelta、およびPDelta法は、用いる予測が、例えば、前のブロック、チャネル、画像、または、これらの方法を用いる前に決定され宣言される他の一式の値である場合に、用いると有利である。したがって、符号化処理されている値と予測値との間の差分は、一度だけで容易に決定され、その値または差分の予測は、符号化処理されている他の値に依存しない。この種の予測手段は、特に量子化との併用にも非常に適している。なぜなら、そのような場合において差分値の量子化は、復号される任意の独立したデータ値ひとつだけに影響し、量子化によって生じる誤差は、復号される他のデータ値に累積しないためである。

【００３８】

多くの場合、用いられる予測、または場合によっては用いられる量子化に関する情報をメソッド選択情報と共に提供してもよい。量子化に関する情報は、１つの量子化値または１つの品質値を用いることによって、データの列全体に対して一度に提供でき、有利であることが多い。そのようなメソッドの例として、IDeltaBlockFromChannel0、PDeltaChannelR_2、DDeltaFrame_4、ODeltaBlockMode、DDeltaPacketPrevと便宜上呼ばれるものなどがある。これらのメソッドのうちの１つ目、すなわちIDeltaBlockFromChannel0は、対応するブロック内のチャネル０の値に関する正の差分値のみを提供する。すなわち、現在符号化処理されているデータブロックの各データ値は、予測として用いられるブロックのチャネル０内の対応する位置にある値より大きいか、または少なくとも等しい。

【００３９】

これらのメソッドの２つ目、すなわちPDeltaChannelR_2は、基礎値に続いて正の差分値、すなわち現在のチャネルとＲチャネルとの差分を、さらに２で量子化したものを提供する。メソッドDDeltaFrame_4は、例えば、徐々に暗くなる画像によく適している。このメソッドは、前のフレームと比較した負の値の変化を、４で量子化したものを提供する。メソッドODeltaBlockModeは、現在のブロック領域をモード値と比較して、ラップされた差分値を提供する。このモード値を各ブロックに個別に送信／提供しても、あるいは、モード値を予測する際に、データチャネル全体のモード値またはデータフレーム全体のモード値を用いてもよい。

【００４０】

DDeltaPacketPrevメソッドは、現在のパケットを同様のサイズの前のパケットと比較して、負のデータ値を提供する。

【００４１】

前記方法は、前記入力データ（Ｄ１）を符号化して前記符号化データ（Ｅ２）を生成する際に、前記エンコーダに少なくとも１つの量子化アルゴリズムを利用させ、前記少なくとも１つの量子化アルゴリズムによって、前記入力データ（Ｄ１）の非可逆的符号化を前記エンコーダに実行させる構成を含んでもよい。

【００４２】

前記方法は、前記入力データ（Ｄ１）に存在する、相互に異なるデータ構造を持つデータを符号化するために、前記エンコーダに相互に異なるアルゴリズムを利用させる構成を含んでもよい。

【００４３】

前記方法は、前記入力データ（Ｄ１）をブロック単位、パケット単位、チャネル単位、ビュー単位、またはフレーム単位で符号化する際に、前記エンコーダ（１００）にＲＤ最適化を利用させる構成を含んでもよい。また、前記方法において、前記ＲＤ最適化は前記エンコーダ（１００）内で計算され、以下の式の値Ｖを最小化してもよく、

Ｖ＝Ｄ＋λ＊Ｒ

ここで歪み（Ｄ）は、前記入力データ（Ｄ１）と、前記符号化データ（Ｅ２）に符号化されて前記復号データ（Ｄ３）に復号された前記入力データ（Ｄ１）の表現との二乗誤差（ＳＥ）の和であり、量（Ｒ）は、例えばビットとして測定される符号化データの量である。

【００４４】

前記方法において、前記少なくとも１つの符号化および／またはエントロピー符号化アルゴリズムは、ＤＣ法、スライド法、マルチレベル法、離散コサイン変換（ＤＣＴ）法、ライン法、スケール法、データベース法、レンジ符号化、ハフマン符号化、連長（ＲＬＥ）符号化、順次連長（ＳＲＬＥ）符号化のうち少なくとも１つを利用するように動作してもよい。

【００４５】

前記方法は、前記エンコーダ（１００）に、ＹＵＶチャネルとＢＧＲチャネルとの少なくとも１つに対応するデータ構造を含む前記入力データ（Ｄ１）を符号化させる構成を含んでもよい。また、前記方法は、前記エンコーダ（１００）に、前記チャネルのデータをＹ、Ｕ、Ｖ、またはＧ、Ｂ、Ｒの順序で符号化させる構成を含んでもよい。アルファチャネル、すなわち透明チャネルを、他のチャネルとは個別に、または一緒に符号化処理することもできる。同様に、符号化処理するデータは音声データであってもよく、その場合、例えばビット深度８、１６、または２４である音声振幅値をチャネルごとに個別に、または多数のチャネルで一度に符号化処理することもできる。したがって、符号化処理するデータが音声、画像、動画、ゲノムデータ、測定結果、テキスト、バイナリなどであるかどうかにかかわらず、符号化処理するデータのビット深度は、例えば、データ要素につき１から２５６ビットまで様々であってもよい。

【００４６】

前記方法は、前記エンコーダに、前記入力データ（Ｄ１）を符号化して前記符号化データ（Ｅ２）を生成するために前記エンコーダが利用した１つ以上の符号化アルゴリズムを示すデータを前記符号化データに含めさせる構成を含んでもよい。

【００４７】

第３の態様によると、符号化データ（Ｅ２）を復号して対応する復号データ（Ｄ３）を生成するデコーダであって、前記第１の態様に従うエンコーダに実装される符号化アルゴリズムの逆を実行するように動作するエンコーダが提供される。

【００４８】

したがって、符号化データ（Ｅ２）を復号して対応する復号データ（Ｄ３）を生成するデコーダであって、少なくとも１つのエントロピー復号アルゴリズムを前記符号化データ（Ｅ２）に適用して処理して、処理済みデータを生成し、Delta復号と共に１つ以上の予測子を用いて、前記処理済みデータを復号して、前記復号データ（Ｄ３）を生成するように動作し、前記１つ以上の予測子は、

（ｉ）　１つ以上の時間的予測子、

（ｉｉ）　複数の値を用いる、１つ以上のローカル空間的予測子、

（ｉｉｉ）量子化の対象となる、１つ以上のローカル空間的予測子、

（ｉｖ）　事前計算された値を用いる、１つ以上のローカル空間的予測子、

のうち少なくとも１つを含むことを特徴とするデコーダが提供される。

【００４９】
事前計算された値について、これらの値は、前述のデータ処理を実行する前に空間的位置に基づいて決定できる任意の値であってよい。すなわち、これらの値は、付属書類１に記載するように、ODelta符号化におけるようなローカル空間的予測子ではない。このような事前計算された値には、例えば、符号化する現在のデータブロックに対して左側または上にある前のブロック、任意の前のチャネル、ビュー、またはフレーム内にある同じデータブロックの位置、任意のデータフレーム内の内部動き予測などがある。

【００５０】
前記デコーダは、方法を示す情報を受信した後、その方法に従って予測を実行し、様々な異なる方法に従って復号する値を計算するように動作してもよい。すなわち、用いられた方法に関する情報がデコーダで受信される。その情報に従って、基礎法を用いるかどうか（PDelta）、正の差分値のみが生じるかどうか（IDeltaまたはPDelta）、負の差分値が生じるかどうか（DDelta）、またはそれらの負の値と正の値を区別するために通常のラップと制限値を用いるかどうか（ODelta）を識別できる。

【００５１】
動作において、前記デコーダは、方法を示す情報を受信し、そのメソッドに従って予測を実行し、様々な異なるメソッドに従って復号する値を計算してもよい。すなわち、用いられたメソッドに関する情報が受信され、その情報に従って、基礎法を用いるかどうか（PDelta）、正の差分値のみが生じるかどうか（IDeltaまたはPDelta）、負の差分値が生じるかどうか（DDelta）、またはそれらの負の値と正の値を区別するために通常のラップと制限値を用いるかどうか（ODelta）を識別できる。予測は一度だけで決定しても、ODeltaと同様に（付属書類１を参照）処理中に決定してもよい。処理中に予測を決定する元のODeltaタイプの手順を、「ローカル空間的予測」または「ローカル予測」と呼ぶ。ブロックの予測が事前に一度だけで決定される前述の新しい予測手段を、「時間的予測」または「空間的予測」と呼ぶ。前述のブロック、チャネル、ビュー、およびフレームに加えて、動き推定も「時間的予測」とみなされる。

【００５２】
第４の態様によると、デコーダにおいて符号化データ（Ｅ２）を復号して対応するデータ（Ｄ３）を生成する方法であって、前記第２の態様に従う方法の逆を前記デコーダにおいて実行することを含む方法が提供される。

【００５３】
デコーダ（１２０）において符号化データ（Ｅ２）を復号して対応するデータ（Ｄ３）を生成する方法であって、少なくとも１つのエントロピー復号アルゴリズムを前記符号化データ（Ｅ２）に適用して処理して、処理済みデータを生成し、Delta復号と共に１つ以上の予測子を用いて、前記処理済みデータを復号して、前記復号データ（Ｄ３）を生成することを含み、前記１つ以上の予測子は、
（ｉ）　１つ以上の時間的予測子、
（ｉｉ）　複数の値を用いる、１つ以上のローカル空間的予測子、
（ｉｉｉ）量子化の対象となる、１つ以上のローカル空間的予測子、
（ｉｖ）　事前計算された値を用いる、１つ以上のローカル空間的予測子、
のうち少なくとも１つを含むことを特徴とする方法が提供される。

【００５４】
事前計算された値については、ローカル予測子と同様に、前述されている。

【００５５】
第５の態様によると、第２の態様または第４の態様に従う方法を実行するために、処理ハードウェアを備えるコンピュータ装置によって実行可能なコンピュータ可読命令が格納された非一時的コンピュータ可読格納媒体を含む、コンピュータプログラム製品が提供される。

【００５６】
本発明の特徴は、添付の特許請求の範囲に定義される本発明の範囲を逸脱することなく

、様々に組み合わせ可能であることを理解されたい。

【図面の簡単な説明】

【００５７】

以下、いくつかの実施形態を、単なる例示として、かつ添付の図面を参照して説明する。

【図１】本願の実施例における上位構造の例の概略図である。

【図２】本願の実施例における、６つのブロックを含むチャネルの例の概略図である。

【図３】ブロックおよびそれに関連付けられた構成要素の例の概略図である。

【図４】本願の実施例に従う、予測のための近隣データ値の概略図である。

【図５】本願の開示に従うエンコーダ、デコーダ、およびコーデックの概略図である。

【図６】付属書類１の開示に従って機能するように実施されるエンコーダ及びデコーダを備えるコーデック装置の図である。

【図７】図６のエンコーダにおいて実行される、データを符号化する方法の工程の図である。

【図８】図６のデコーダにおいて実行される、データを復号する方法の工程の図である。

添付図面において下線の引かれた番号は、その番号が位置するアイテムやその番号が隣接するアイテムを表すために使用される。下線が引かれていない番号は、その番号と線で結ばれて特定されるアイテムに関連している。番号に下線が引かれず矢印と共に記されている場合、その番号は矢印が指すアイテム全般を特定するために使用される。

【発明を実施するための形態】

【００５８】

本願の実施例の説明において用いる略語を以下の表１に示す。

【００５９】

表１：実施形態の説明に用いる略語の説明

略語	説明
１Ｄ	１次元。例えば、信号やデータパケットを指して用いる。
２Ｄ	２次元。例えば、信号やデータパケットを指して用いる。
３Ｄ	３次元。例えば、信号やデータパケットを指して用いる。
ブロック	デジタルデータからの複数のデータ要素。すなわちデジタルデータの一部
ＣＲＣ	巡回冗長検査
コーデック	デジタルデータ用のエンコーダおよびデコーダ
ＤＢ	ＲＡＭベースまたはＲＯＭベースメモリ内のデータベース
ＤＣ	画像のＤＣ成分。平均輝度に対応し、画像内の最低空間周波数を示す画像平均
Delta符号化	Delta符号化は、データの格納や転送を完全なデータファイルとしてではなく、連続したデータ同士の差分の形式で行う方式である。
ＩＳＰ	インターネットスイッチプロバイダ
ＬＡＮ	ローカルエリアネットワーク
パケット	複数のデータブロックなどを含むデータのかたまり
ＲＡＭ	ランダムアクセスメモリ
ＲＤ	レート歪み
ＲＬＥ	連長符号化
ＲＯＩ	関心領域
ＲＯＭ	読み出し専用メモリ
ＳＲＬＥ	分割連長符号化
ＶＬＣ	可変長符号
ＸＯＲ	排他的論理和（論理関数）

【００６０】

概要として、本願の実施例は、改良された形式のエンコーダとデコーダ、および関連する改良されたデータ符号化方法とデータ復号方法に関する。本願の実施例は、以下に詳述し、本願の実施例においてさらに改良しているODelta符号化方法などの、Delta符号化方法に基づいている。Delta符号化方法は、音声パケット、画像ブロック、インターネットデータパケット、チャネル、動画フレームなどを、相互に異なる多様な空間的および時間的予測方法を用いて符号化処理するために提供され、任意で量子化子を利用してもよい。本願の符号化方法は、可逆的および非可逆的な符号化の両方に適している。これらの符号化方法は、次に示す３つの主な機能的要素を含む。

（ｉ）　予測

（ｉｉ）　ODelta演算子、またはPDelta、IDelta、DDeltaなどの類似の演算子、ならびに任意使用の量子化子

（ｉｉｉ）エントロピー符号化

【００６１】
本願の符号化方法はエンコーダにおいて用いることができ、対応する復号方法はデコーダにおいて用いることができる。これについては、図５を参照して詳細に後述する。
【００６２】
添付の付属書類１に、ＤＰＣＭ式の使用方法に適したODelta演算子の説明を記載している。付属書類１より前の本明細書の本文において、これらのODelta演算子は、様々な（ローカル）空間的予測方法、時間的予測方法、またはそれらを組み合わせた方法を用いるために修正されている。本願の方法は、データ列全体、個々のデータフレーム、個々のデータチャネル、個々のデータブロック、個々のデータパケットなどに用いることができるように考案されている。また、本願の方法は、選択された予測方法、選択されたODelta演算子、および選択された残差符号化方法、圧縮方法に基づいて、相互に異なる複数の符号化処理方法を提供する。
【００６３】
本願の方法と組み合わせて他の多くの符号化処理方法を用いてもよい。これらの方法は、付属書類２に示すブロックエンコーダおよび付属書類３に示すブロックデコーダと共に用いると有利である。前者は英国特許第２５０３２９５号に記載され、参照により本出願に組み込まれる。後者は英国特許第２５０５１６９号に記載され、参照により本出願に組み込まれる。任意のデータブロックの符号化に最適な符号化処理方法は、例えば、任意のデータチャネルがブロック単位で符号化処理される場合、例えばＲＤ最適化を用いて選択される。ＲＤ最適化によって、以下の式における値Ｖが最小になる。
Ｖ＝Ｄ＋λ＊Ｒ　　式１
ここで、歪み（Ｄ）は典型的に、元の値と復号された値との二乗誤差（ＳＥ）の和であり、符号化されたデータ値の量（Ｒ）は典型的にビットで測定される。本願の開示に従う方法において、他の多くの符号化処理方法を用いてもよい。これには例えば、ＤＣ法、スライド法、マルチレベル法、ＤＣＴ法、ライン法、スケール法、データベース法などがある。
【００６４】
本願の開示に従う方法は、符号化する入力データ（Ｄ１）内に存在する相互に異なるデータ構造に有利に用いられる。例えば、データチャネル全体、例えば平面画像の輝度チャネルを、本願の開示に従う方法によって符号化処理してもよい。本願の開示に従う方法は単純であり、例えば、携帯電話やカメラなどの低消費電力の携帯電子機器に近年用いられる縮小命令セット（ＲＩＳＣ）プロセッサなどを用いる機器やシステムを複雑にすることなく実行できる。このため、チャネル全体に対して本願の方法を実行した結果を、黒／モード値チャネル、フリーズチャネル、エントロピー符号化された元のチャネル、ブロックエンコーダによるチャネル符号化などの他のチャネル符号化方法と容易に比較できる。最適なチャネル符号化方法は、ＲＤ最適化によっても有利に選択される。ブロックエンコーダによるチャネル符号化方法とは、相互に異なる符号化処理方法によってデータチャネルをブロック単位で符号化処理することであり、この場合、データチャネル全体に対して１つの符号化処理方法を用いるわけではない。
【００６５】
本願の方法は、非可逆的符号化において、すなわち量子化された残差値の符号化と共に、または残差の符号化なしで用いてもよく、あるいは、可逆的符号化において、すなわち残差なし、または量子化なしの残差値符号化と共に用いてもよい。エンコーダおよびデコーダにおける前の符号化値および復号値はすべて、本願の開示に従って、現在または将来のデータ値の予測に用いることができる。本願の符号化方法を適用してデータを可逆的に符号化処理する場合、可逆的符号化においては復号値と同じである処理済みのソース値を、エンコーダにおける現在または将来のデータ値の予測に用いることができる。
【００６６】
ODelta演算子の最も重要なパラメータはhighValue、lowValue、およびwrapValue（すなわち、少なくともhighValue－lowValue＋１）である。これについては付属書類１におい

てより詳細に説明する。量子化を利用するようにhighValueおよびlowValueを定義することも可能である。例えば、元データは０から２５５までの値を含むが、選択された品質係数（例えば、品質値が「１」から「１００」の範囲である場合は、３０である。ここで品質値「１００」は可逆的圧縮を指す）により、最終結果を例えば０から７８の値に量子化することが望ましい（相対量子化７８／２５５）。ODelta演算子を用いる前（プリオフセット）または後（ポストオフセット）にデータオフセットを用いることもできる。また、ODelta演算子の後でエントロピー符号化を実行することも有利である。そうしないと、エントロピーの低減がデータ符号化に十分に活用されない。

【００６７】

前述のIDelta、DDelta、およびPDelta法を参照すると、付属書類１に説明されるように、ODelta法において必要であったwrapValueを決定する必要がなくなっている。したがって、前述のIDelta、DDelta、およびPDelta法に関しては、メソッド選択情報および最終的な量子化情報と共に、highValue'およびlowValue'がより重要な役割を担う。この場合、highValue'およびlowValue'は最終的な実際のデータ値の値範囲ではなく、提供される差分値の値範囲を示す。lowValue'はPDelta法の基礎（pedestal）値の決定にも用いられる。この場合、提供される最大データ値は、highValue'－lowValue'の結果であるか、または量子化によって制限される。この量子化は、因数によって、品質パラメータによって、または元のダイナミックレンジに対する相対的な変化として決定することができる。このような実施の例として、前述のように、元のhighValueまたは最大差分highValue'が２５５であり、値範囲が相対量子化値７８／２５５によって制限され縮小されている場合がある。この例においては、相対量子化の量子化値は、例えば、値７８あるいは値０．３０５９（すなわち７８／２５５未満）として提供することができる。

【００６８】

用いるエントロピー符号化方法は、レンジ符号化またはＳＲＬＥレンジ符号化を選択すると有利であるが、例えばハフマン符号化、ＲＬＥ符号化、ＳＲＬＥ符号化などの他のエントロピー符号化方法を用いることもできる。付属書類１において説明するODelta法を用いる場合、予測値は常に前のデータ値であることが望ましく、最初の予測値を、選択された初期化方法によって初期化する必要がある。

【００６９】

本願の開示に従う方法では、予測値は、前のデータ値のみを用いるのとは異なるように選択することもできる。本願の方法に従うと、予測値として、選択された１つのデータ値、または複数のデータ値から算出された１つの値を用いることが可能になる。例えば、この値は、２つ以上の前のデータ値（１Ｄ）、近隣の２つ以上の前のデータ値（２Ｄ、３Ｄ、...）、前のデータブロックまたはデータパケット内の１つまたは複数のデータ値、１つまたは複数の前のデータチャネル内の１つまたは複数のデータ値、１つまたは複数のデータフレーム内の１つまたは複数のデータ値、前述のデータ値の任意の組合せなどから計算できる。

【００７０】

現在のデータ値の予測値が算出されると、元データ値と予測データ値との差分または合算がOValueとして計算される。OValueはQOValueへと量子化、または複製でき、その後ODelta演算子へと渡すことができる。ODelta演算子は、QOValueがlowValueより小さい場合またはhighValueより大きい場合は（wrapValueの加算および／または減算により）ラップアラウンドを実行する。

【００７１】

値が量子化される場合は特に、ラップアラウンドおよび逆ラップアラウンドが正確に実行されるように量子化レベルを設計し、ラッピング演算によってこの方法の結果が正の加算（または減算）から負の加算（または減算）へ、またはその逆へと変わらないようにする必要があることを理解されたい。ラップアラウンドによって結果が明らかに小さい絶対値や明らかに大きい絶対値へと変わることがないようにすべきである。つまり、例えば、絶対データ値に対して２つの異なる量子化子を用いる場合、逆量子化およびラップアラウ

ンドにおけるデータ値の誤解析を回避するために、小さいデータ値と大きいデータ値を中間のデータ値より小さい量子化値によって量子化する必要がある。

【００７２】

図１を参照すると、フレーム、ビュー、チャネル、データブロック、データパケット、個々のデータ値などの異なるデータ構造に分割された３Ｄ動画コンテンツが図示されている。これに加えて、フレームのグループ、データブロックのグループ、initデータブロック、データスライスなどの構造を用いてもよい。本願の開示に従うと、入力データ（Ｄ１）を処理および符号化して対応する符号化データ（Ｅ２）を生成する際に、このような相互に異なる構造をすべて分離しなくてもよい。データの順序は異なってもよいが、図１の例では、任意のチャネル内のブロックは左から右へ、かつ上から下へと処理される。処理済み、すなわち符号化済み、および任意で非可逆的符号化を実行する場合は復号済みの値はすべて、現在および将来の値の予測に用いることもできる。そうすることで、本願の開示に従うエンコーダおよび対応するデコーダの両方が、符号化値から復号された値に関する情報を持つことができるからである。

【００７３】

本願の開示に従って、ＹＵＶチャネルをＹ、Ｕ、Ｖの順序で符号化し、かつ、ＢＧＲチャネルをＧ、Ｒ、Ｂ、またはＧ、Ｂ、Ｒの順序で符号化して、本願の開示に従う方法における時間的チャネル予測のより良い選択肢を利用可能にすると有利であることが多い。時間的チャネル予測は、一般に、任意の画像がＲＧＢ色空間へと符号化される場合に非常に優れた方法である。時間的チャネル予測では、チャネル相関が大幅に低減される。ＹＵＶ色空間を用いる場合、この色空間の特性により、チャネル相関は既に大幅に低減されている。また、ＹＵＶ色空間を用いる場合、情報の大部分がＹチャネルに含まれる。つまり、ＵチャネルとＶチャネルをより効率的に符号化処理できる。

【００７４】

本願の方法は、相互に異なるデータ構造の処理に用いることができる個別のサブメソッド、すなわちアルゴリズムを含んでもよい。例えば、色チャネルは空間的予測または時間的予測によって符号化処理できる。様々な種類の時間的予測のサブメソッドを用いることもできる。あるサブメソッドは、例えば、前のフレーム内と同様の位置にある同じチャネル値を用い、別のサブメソッドは、前のビュー内と同様の位置にある同じチャネル値を用い、さらに別のサブメソッドでは、例えば、チャネル０からの同様の位置にある値を用いて、例えばチャネル２の値を予測する。

【００７５】

時間的予測を用いる場合、最初の予測値を定義する必要はない。現在のチャネル内の各値に、他のフレーム、ビュー、またはチャネルにおいて利用可能な予測値が含まれる。「時間的な予測」は、任意のチャネル内で空間的に前に位置する類似のデータブロックに用いることもできる。一般に、この種の方法を用いる場合、利用可能なデータブロックの選択肢は限られているため、異なるサブメソッドの数が多くなりすぎることはない。つまり、「動きベクトル」のようなブロック記述子を送信する必要もない。この種のデータブロック記述子を用いてもよいが、一般に、記述子の精度は、例えば、データブロック単位であり個々のデータ値単位ではないため、イントラ動きベクトル推定を用いる場合と比べて、異なる組合せの数が大幅に少なくなる可能性がある。

【００７６】

一般に知られているイントラ／インター動き推定方法によって得られるものと同等の精度を持つ、利用可能な代替アルゴリズムの組合せを用いたとしても、本願の方法の選択肢には、0Delta符号化の利用による効率的な残差符号化が含まれるため、本願の開示に従う方法の利用にはやはり大きな利点がある。したがって、本願の開示に従う方法では、予測のために個別にイントラ／インター動き推定を用いた後、残差符号化のために別途、例えばＤＣＴ法を用いるという必要がない。

【００７７】

本願の実施例では、空間的予測において相互に異なる予測値を用いてもよい。前の値の

予測（Ｘに対するＡ）は、ODelta技法からの既知の技術である（付属書類１を参照）。本発明の実施形態において利用すると有利な方法の１つでは、値Ｘに対してＰ＝Ａ＋Ｂ－Ｃなどの予測値を用いる。例えば、２Ａ－DまたはPNG文書に記述されているPAETH予測などの他の多くの予測値を用いてもよい。予測値は、可能な値の範囲になるように制限、または切り捨てると有利である。例えば、チャネル０における任意の値が、０（lowValue）と６３（highValue）の間で、Ａ＝６０、Ｂ＝６１、Ｃ＝５２という値として提供される可能性がある場合、Ｐ＝Ａ＋Ｂ－Ｃ＝６９であり、この値は値６３（＝highValue）になるように切り取られるか、切り捨てられるか、または飽和させられる。

【００７８】

Ｘ＝６２の場合、修正されたODelta演算子方法１によるOValueは、６２－６３＝－１である。この値は、可逆的符号化においては量子化する必要がないため、QOValueも－１である。ここで、－１はlowValue（０）より小さいので、６３をODelta値として取得するにはwrapValue（６４）をQOValueに加える必要があることを理解されたい。この値は、このChannelSpatialODeltaCodedメソッドによって値Ｘを符号化するために、エントロピー符号化のためのバッファに設定される。他のデータ値に対しても同様の処理が行われ、すべてのチャネル値が処理されたら、ODelta値のバッファの内容が、例えば、レンジ符号化、または順次連長符号化（ＳＲＬＥ）レンジ符号化などによって圧縮され、チャネル０に対する出力符号化データ値が作成される。

【００７９】

予測値としてＡ＋Ｂ－Ｃを用いる場合、最初１の行が処理されるとき、予測に使用可能な値はＡのみであるため、値Ａが予測値として直接用いられることを理解されたい。同様に、最初の列に対して使用可能であるのはＢ値のみであるため、値Ｂが予測値として直接用いられる。最初の値、例えばチャネル０内の最も左上側の値は、予測に使用可能な空間値を持たない。例えば、前のビューまたは前のフレームに時間値があれば、その時間値を最初の予測値に用いることができる。適切な時間予測値が使用可能でない場合、例えば、値０、または中点値（（６３－０＋１）÷２＝３２）、または別途提供されたチャネル／ビュー／フレームのモード値を、チャネル０に対する最初の予測値として用いることができる。１つの方法において複数の予測子を用いて、符号化データ（Ｅ２）内のどのフレーム、チャネル、ブロック、さらにはデータ値にどの予測子を用いるのかに関する情報を提供することも可能である。

【００８０】

同様の方法を、例えばデータブロックに用いることもできる。以下の例に、BlockChannel00DeltaCodedメソッドを用いてチャネル２内のブロック２を符号化処理する方法を示す。BlockChannel00DeltaCodedメソッドでは、品質要件が明らかに低い、値４による量子化が対応する符号化値に用いられる。値４による量子化とは、lowValueが０、highValueが１５、およびwrapValueが１６であることを意味する。ここで、現在のチャネル２内のブロック２には、例えば以下のような値が含まれている。

４５，４８，５０，５２
４６，４８，５０，５１
４６，４９，４９，５０

【００８１】

チャネル０内のブロック２には、以下の予測値が含まれている。

３６，３９，４０，４２
３６，３７，３９，４１
３６，３９，３９，４１

【００８２】

時間的予測を用いるため、符号化値を量子化すると、チャネル２の復号値は変更されるが、チャネル０内の予測値には影響しない。このため、OValueを定義する場合に量子化を

考慮する必要がないため、処理を単純化できる。その後、0Valueは以下のようになる。

９，９，１０，１０，１０，１１，１１，１０，１０，１０，１０，９

【００８３】

これらの値を値４で除算することによって量子化すると、以下のようなQ0Valueが生成される。

２，２，２，２，２，２，２，２，２，２，２，２

【００８４】

すべての値が範囲内、すなわち０から１５の値であるため、どのQ0valueにもラップアラウンドは必要なく、0Delta値がQ0Valueと同じになる。ここで、すべての0Delta値が同じであるため、用いる符号化処理メソッドをBlockChannel00DeltaCodedメソッドからBlockChannel00DeltaSameメソッドまたはBlockChannel0IDeltaSameメソッドへと変更することができることを理解されたい。このBlockChannel00DeltaSameメソッドでは１つの値（２）のみが提供される必要がある。それによって、ブロック値の復号が、デコーダにおいて適切に、すなわちエンコーダにおいて実行されたのと同様の方法で実行される。

【００８５】

このデータブロックに対し、デコーダにおいて符号化処理メソッドBlockChannel00DeltaSameおよび値２を用いるように指示される。その後デコーダは、以下のような１２（４×３ブロック）個の値を含むバッファを作成する。

２，２，２，２，２，２，２，２，２，２，２，２

【００８６】

この例では、逆量子化によってこれらの値に４を掛ける。また、量子化範囲内の誤差をより正確に推定するために、１を加算してもよい。これらの値を逆量子化すると、以下のような値が生成される。

９，９，９，９，９，９，９，９，９，９，９，９

【００８７】

これらの逆量子化値を、エンコーダにおけるチャネル０からの値に類似する予測値に加算すると、ブロック値が以下のようになる。

４５，４８，４９，５１
４５，４６，４８，５０
４５，４８，４８，５０

【００８８】

符号化および復号における値の歪み、すなわち、符号化／復号処理後の任意の元データと対応する復号データとの間の誤差は、以下のようになる。

０，０，１，１
１，２，２，１
１，１，１，０

【００８９】

この歪みは量子化によるものであるが、非常に小さい。この方法では、任意の４×３データブロック全体に対して提供する必要があるのは選択された方法と１つ値のみであるため、非常に効率的である。チャネル内でより多くのブロックを、同じ、または実質的に同様の符号化方法によって符号化処理する場合、それらすべての値を、例えば、任意のエンコーダにおいてレンジ符号化し、対応する任意のデコーダにおいてレンジ復号することで、より高い圧縮率、すなわち、元データまたは復号データ（Ｄ１）の量／符号化データの量（Ｅ２）の比率を達成できる。

【００９０】
前述のODeltaを用いる場合、チャネル０より前にチャネル２を符号化し、チャネル０においてチャネル２を予測のために用いても有利である。この場合、すべてのQOValueが－２であると有利である。これらの値はlowValueよりも小さいため、ラップアラウンド（すなわちwrapValueの加算）が必要である。その後、すべてのODelta値が１４（－２＋１６）と等しくなる。したがって、本願の実施例を実行するために利用される前述のODelta符号化は、範囲を拡大することも、符号ビットを必要とすることもないため、同様の既知のDelta符号化方法よりも効率的である。また、ODelta符号化は、この現在のデータブロックの差分チャネルに対してＤＣ法を用いるよりも効率的である。
【００９１】
次に、チャネル１において４×３ブロックをBlockSpatialODeltaCodedによって空間的に符号化する例を示す。この例において、このメソッドは最初の予測値として０を用いる。元の値は０から２５５の範囲であり、品質は、lowValueが０、highValueが３５、wrapValueが３６になるようにエンコーダにおいて設定されている。もちろん、対応する復号段階において、値が元の範囲の０から２５５内のままであり、したがって復号段階においてlowValueが０、highValueが２５５、wrapValueが２５６であると有利である。この場合、元のブロックは以下のとおりである。

１４１，１５１，１４８，１３７
１５９，１５０，１５２，１４７
１５９，１５４，１５３，１５０

【００９２】
最初の１６レベル、すなわち、レベル０から１５、および最後の１６レベル、すなわちレベル２０から３５、に対する量子化子は７であり、中間の４レベル、すなわちレベル１６から１９に対する量子化子は８である。これは、１１２より小さい絶対差分値、すなわち０から１１１はすべて値７によって量子化される、すなわち除算されることを意味する。後続の１４４までの値、すなわち１１２から１４３は、値１１２が減算され、値８によって量子化され、値１６が加算される。最後の値、すなわち１４４から２５５は、値１４４が減算され、値７によって量子化され、値２０が加算される。同様に、最初の１６個の値、すなわち値０から１５の逆量子化は、各値に値７を乗算することによって行われる。中間の値である１６から１９は、各値から値１６を減算した後、各値に８を乗算して値１１２を加算することで復号される。最後の値、すなわち値２０から２５は、各値から値２０を減算した後、それらに値７を乗算し値１４４を加算することで復号される。
【００９３】
第１の値は１４１である。予測値は０に初期化されたため、OValueは１４１、QOValueは１９となる。ラップアラウンドは不要であるため、ODelta値も１９となる。この値はラップアラウンドなしで値１３６（１６＊７＋３＊８）に復号される。この値を将来の予測に有利に用いてもよい。
【００９４】
第２の値は１５１である。Ａに基づく予測値を用いてもよく、この値は１３６である。OValueは１５であり、QOValueは２である。ここでもラップアラウンドは不要であるため、ODelta値は２である。この値はラップアラウンドなしで値１５０（１３６＋２＊７）に復号される。この値を将来の予測に有利に用いてもよい。
【００９５】
第３の値は１４８である。Ａに基づく予測値はここでは１５０である。OValueは－２であり、QOValueは０である。ここでもラップアラウンドは不要であり、OValueは負の値であるが、量子化された値QOValueは０であり、範囲内であるため、ODelta値も０である。この値はラップアラウンドなしで値１５０（１５０＋０）に復号される。この値を将来の予測に有利に用いてもよい。
【００９６】

第４の値は１３７である。Ａに基づく予測値はここでも１５０である。0Valueは－１３、Q0Valueは－１である。ここではラップアラウンドが必要であり、0Delta値は－１＋３６＝３５である。この値が復号されると、値１５０＋２４９はhighValueすなわち値２５５より大きいため、ラップアラウンドが必要であり、その結果は１５０＋２４９－２５６＝１４３である。この値を将来の予測に有利に用いてもよい。

【００９７】

第５の値が新しい行に入力され、この値は１５９である。ここで、値Ｂ，すなわち１３６を用いて予測を実行できる。0Valueは２３であり、Q0Valueは３である。ラップアラウンドは不要であるため、対応する0Delta値も３となる。復号値は１３６＋２１＝１５７となる。この値を将来の予測に有利に用いてもよい。

【００９８】

第６の値は１５０であり、予測値としてＡ＋Ｂ－Ｃを用いることができる。予測値はここで１５７＋１５０－１３６＝１７１である。0Valueは－２１であり、Q0Valueは－３である。ラップアラウンドが必要であるため、0Delta値は３３である。復号値もラップアラウンドする必要があり、その結果は１５０（１７１＋２３５－２５６）である。この値を将来の予測に有利に用いてもよい。

【００９９】

同様に、任意のデータブロックの最後まで有利に処理が続けられ、全体的な0Delta符号化結果は以下のとおりとなる。

１９，２，０，３５，３，３３，０，０，０，０，０，１

【０１００】

これらの値はバッファに有利に挿入され、他の同様の符号化0Delta値と共に、例えばＳＲＬＥレンジ符号化を用いてエントロピー符号化される。デコーダは、符号化データ（Ｅ２）内のメソッドBlockSpatial0DeltaCodedに関する情報および符号化0Delta値が提供されると、以下の値を生成することができる。

１３６，１５０，１５０，１４３
１５７，１５０，１５０，１４３
１５７，１５０，１５０，１５０

【０１０１】

符号化および復号後の値の歪み、すなわち、用いられた符号化と復号が相互に逆である場合の元データ（Ｄ１）と復号データ（Ｄ３）との間の誤差は、以下のとおりである。

５，１，－２，－６
２，０，２，４
２，４，３，０

【０１０２】

第１の値は任意の値であってもよく、優れた初期化推定を有利に利用して結果を容易に改善できることを理解されたい。また、これらの第１の値を個別に提供し、他の値の符号化結果を改善することもできる。一般に、他の値には、多数のゼロ、および少数の１に近い値、＋１に対応する値、３５に近い値、－１に対応する値が含まれる。この例では、highValueは３５、wrapValueは３６であった。

【０１０３】

符号化処理されたすべての値がゼロであることもあるが、その場合、例えば、BlockSpatial0DeltaNotCodedメソッドを用いると有利である。BlockSpatial0DeltaNotCodedメソッドは空間的予測のみを行い、0Delta符号化のための値は提供しない。一般に、空間的予測には符号化処理された値が必要であるが、時間的予測は、すべて符号化された値すなわち「Coded」メソッドの代わりに、定数値すなわち「Same」を用いるか、または符号化処理された値なし、すなわち「NotCoded」を用いて実行されることが多い。

【０１０４】
本願の開示に従うメソッドを用いる場合、それらのメソッドは一般に、相互に異なるメソッドとして用いられる。基礎となる何らかのメソッドを用いて、例えば、用いられる任意の時間的予測ソース、符号化処理手段などを記述するサブメソッドを利用することもできる。次の例において、表２に、相互に異なるチャネルおよびブロックの符号化サブメソッドの一覧を示す。各サブメソッドは、関連付けられた３ビットによって表現、すなわち定義される。

【０１０５】
表２：チャネルおよびブロックの符号化処理メソッド

メソッド	基礎となるメソッド	サブメソッド
Channel_Channel00DeltaCoded	Channel0Delta	000
Channel_Channel10DeltaCoded	Channel0Delta	001
Channel_ViewPreviousChannel00DeltaCoded	Channel0Delta	010
Channel_ViewPreviousChannel10DeltaCoded	Channel0Delta	011
Channel_ViewPreviousChannel20DeltaCoded	Channel0Delta	100
Channel_FramePrevious0DeltaCoded	Channel0Delta	101
Channel_ChannelPrevious0DeltaNotCoded	Channel0Delta	110
Channel_Spatial0DeltaCoded	Channel0Delta	111
Block_Channel00DeltaCoded	Block0Delta	000
Block_Channel10DeltaCoded	Block0Delta	001
Block_Channel00DeltaSame	Block0Delta	010
Block_Channel00DeltaNotCoded	Block0Delta	011
Block_FramePrevious0DeltaCoded	Block0Delta	100
Block_BlockLeft0DeltaCoded	Block0Delta	101
Block_BlockUp0DeltaCoded	Block0Delta	110
Block_Spatial0DeltaCoded	Block0Delta	111

【０１０６】
異なるチャネルの符号化処理に、相互に異なる符号化処理メソッドを用いてもよい。例えば、任意の第１のチャネル（チャネル０）を、データブロックの符号化に任意でBlock0Deltaメソッドも用いるブロックエンコーダによって符号化し、任意の第２のチャネル（チャネル１）を、時間的予測ベースの0Delta法であるChannel_Channel00DeltaCodedによって符号化し、任意の第３のチャネル（チャネル２）を、空間的予測ベースの0Delta法であるChannel_Spatial0DeltaCodedによって符号化する。すべてのチャネルを同じ方法、例えば空間的予測ベースの0Delta法で符号化処理する場合、例えばFrame_Spatial0DeltaCodedメソッドを、対応するフレームに有利に提供すれば、そのフレームに他の符号化処理方法を提供する必要はない。すなわち、チャネル符号化方法やブロック符号化方法を用いる必要はない。

【０１０７】
本願の実施例に関連するデータ構造を、図１から図４を参照して以下に説明する。本願

の方法における空間的予測に用いることができるデータ構造およびデータ値を示している。図１に、上位構造の例を示す。この例のデータは、例えば３つのフレームを含み、各フレームには例えば２つのビューがあり、各ビューには例えば３つのチャネルがある。上位構造のデータには、スライス、データブロックのグループ、または他のデータ構造が含まれてもよい。一部の構造、例えばパケットおよび／またはビューが含まれていなくてもよい。存在する構造の違い、または値の処理順序や利用する構造の選択により、データ値の予測は大幅に異なる可能性がある。図２に、６つのブロックを含むチャネルの例を示す。これらのブロックの符号化順序は、左から右、上から下である。図２のブロックは例えば３つのパケットを含み、各パケットは例えば４つの値を含む。また、図３に、ブロックとその構成要素の例を示す。

【０１０８】

したがって、全体的なデータ例は、データ＝３＊２＊３＊６＊３＊４個の値＝１２９６個の値となる。次の例は、予測に用いられる近隣のデータ値を表す。図４において、値ＡからＮは、位置Ｘに対する前のデータ値である。値ｏから値ｔは、例えば、任意の現在のブロックより前に処理された１つ以上の他のブロック内にある場合、前のデータ値である。前のデータパケット、データブロック、チャネル、ビュー、およびフレームも、予測に使用可能なデータ値を含む可能性がある。したがって、図４は、本願の開示に従って予測のために使用可能な近隣のデータ値の概略図である。

【０１０９】

前述の本願の実施例は、前述のとおり、ODelta演算子の修正版を用いてデータ圧縮結果を向上させることができる方法を提供する。本願の方法は、相互に異なり多数になり得る予測方法の選択肢と、ODeltaラッピングに対して適切に動作する任意の量子化子とを用いる。本願の方法は、ODelta演算によるエントロピー低減によりもたらされるすべての利点を活用するために、エントロピー符号化も用いる。これらの方法は、フレーム、チャネル、データブロック、データパケットなどの相互に異なる多様なデータ構造に適している。各構造には、少量の対応するODelta符号化データを作成する、明確に定義された符号化方法が利用される。符号化データの格納および通信に必要なデータ通信帯域幅は、そのデータが符号化されない場合と比べて大幅に削減される可能性があり、非常に有利である。

【０１１０】

前述の方法および実施形態は、データエンコーダおよびデータデコーダに関して有利に実行される。図５を参照すると、本願の実施例は以下に関する。

（ｉ）　入力データＤ１を符号化して対応する符号化データＥ２を生成するエンコーダ１００、および、前記入力データＤ１を符号化して前記符号化データＥ２を生成するための、対応する方法。

（ｉｉ）　前記符号化データＥ２を復号して対応する復号データＤ３を生成するデコーダ１２０。前記復号データＤ３は、可逆的符号化におけるように、前記入力データＤ１と厳密に類似であってもよく、あるいは、非可逆的符号化におけるように、前記入力データＤ１と略類似であってもよい。あるいは、前記データＤ３は、例えば変換によって、前記入力データＤ１と異なってもよいが、前記入力データＤ１に存在する情報と、前記符号化データＥ２を復号して前記復号データＤ３を生成するための対応する方法を、実質的に維持しているものとする。

（ｉｉｉ）少なくとも１つのエンコーダ１００と少なくとも１つのデコーダ１２０との組合せを含むコーデック１３０。前記コーデック１３０は、単一の機器内に実装されても、複数の機器間で有効に実装されてもよい。例えば、前記コーデック１３０は、第１の空間的位置にエンコーダ１００があり、他の複数の空間的位置に複数のデコーダ１２０があるブロードキャストシステムとして実装されてもよい。

【０１１１】

添付の特許請求の範囲に定義される発明の範囲を逸脱することなく、前述した本発明の実施形態への修正が可能である。本発明の記述と特許請求の範囲で用いられる「含む」、「備える」、「包含する」、「構成される」、「有する」、「存在する」などの表現は、

包括的構成であると解釈されることを意図しており、明示的に記載されていないアイテムや部品、構成要素も含まれ得ることを意図している。単数による表記は、複数の場合も関連すると解釈される。添付の特許請求の範囲における括弧内の数字は、請求項の理解を助けることを意図したものであり、これらの請求項によって定義される発明の範囲を限定するように解釈されるべきではない。

〔付属書類１：0Delta符号化の概要〕

【０１１２】

以下に0Delta符号化の概要を述べる。0Delta符号化及び関連技術の説明のために、エンコーダ１０１０、エンコーダ１０１０を使用する方法、デコーダ１０２０、デコーダ１０２０を使用する方法を例示する。

【０１１３】

エンコーダ１０１０は、数値列を含む入力データ（ＤＡ１）を符号化することにより、対応符号化出力データ（ＤＡ２又はＤＡ３）を生成するエンコーダである。このエンコーダは、差分符号化及び／又は合算符号化の方式を入力データに適用することにより１つ又は複数の対応符号化列を生成するデータ処理装置を備える。但し、上記１つ又は複数の対応符号化列は、符号化出力データを生成するために、最大値におけるラップアラウンド及び／又は最小値におけるラップアラウンドを受ける。またエンコーダ１０１０は、符号化出力データを作成するのに利用される一連の予測値として、デフォルト第１予測値を利用するように動作可能である。符号化出力データ（ＤＡ２又はＤＡ３）は、入力値、予測値及び符号化演算を用いて生成される。

【０１１４】

エンコーダ１０１０を使用する方法は、数値列を含む入力データ（ＤＡ１）を符号化するエンコーダを用いることにより、対応符号化入力データ（ＤＡ２又はＤＡ３）を生成する方法である。この方法は、

（ａ）差分符号化及び／又は合算符号化の方式を入力データに適用することにより、１つ又は複数の対応符号化列を生成するエンコーダのデータ処理装置を用いることと；

（ｂ）符号化出力データを生成するために、上記１つ又は複数の対応符号化列に対して最大値におけるラップアラウンド及び／又は最小値におけるラップアラウンドを行う、上記データ処理装置を用いることと；

を含む。

【０１１５】

デコーダ１０２０は、符号化データ（ＤＡ２、ＤＡ３又はＤＡ４）を復号することにより対応復号出力データ（ＤＡ５）を生成するデコーダである。このデコーダは、符号化データの１つ又は複数の部分を処理するデータ処理装置を備え、該データ処理装置は、上記１つ又は複数の部分の１つ又は複数の対応符号化列に対して差分復号及び／又は合算復号の方式を適用するように動作可能である。上記１つ又は複数の符号化列は、復号出力データを生成するために、最大値におけるラップアラウンド及び／又は最小値におけるラップアラウンドを受ける。

【０１１６】

デコーダ１０２０を使用する方法は、符号化データ（ＤＡ２、ＤＡ３又はＤＡ４）を復号するデコーダを用いることにより対応復号出力データ（ＤＡ５）を生成する方法である。この方法は、上記復号データの１つ又は複数の部分を処理するデータ処理装置を用いることを含む。上記データ処理装置は、上記１つ又は複数の部分の１つ又は複数の対応符号化列に対して差分復号及び／又は合算復号の方式を適用するように動作可能であり、上記１つ又は複数の符号化列が、復号出力データを生成するために、最大値におけるラップアラウンド及び／又は最小値におけるラップアラウンドを受ける。

【０１１７】

デコーダ１０２０を使用する方法には、次のような方法もある。この方法は、符号化データ（ＤＡ２、ＤＡ３又はＤＡ４）を復号するデコーダを用いることにより対応復号出力データ（ＤＡ５）を生成する方法であって、該方法は、

（ａ）上記符号化データを処理するものであり、上記符号化データは、変換データの連続値における変化を表す少なくとも１つの符号化列を含み、かつ最大値におけるラップアラウンド又は最小値におけるラップアラウンドを利用するものであることを考慮して、上記符号化データの１つ又は複数の部分に復号を適用するデータ処理装置を用いることと；
（ｂ）対応処理データを生成し、かつ少なくとも１つのプリオフセット値及び／又はポストオフセット値を用いて上記１つ又は複数の部分を変換することにより復号出力データを生成するデータ処理装置を用いることと；
を含む。

【０１１８】

図６～図８を参照して、付属書類１に記載の実施形態を例としてのみ以下に説明する。

【０１１９】

本開示の実施形態を説明する際、表３に示す通り以下の頭字後及び定義が用いられる。

表３：頭字語及び定義

頭字語	説明
ADC	アナログ・デジタル変換器（Analog-to-digital converter）
Codec	デジタルデータに関する符号器及び対応復号器
DAC	デジタル・アナログ変換器（Digital-to-analog converter）
DB	ランダムアクセスメモリ（ＲＡＭ）又は読出専用メモリ（ＲＯＭ）におけるデータベース
DC	所与の画像のＤＣ要素、即ち画像の平均値、即ち平均輝度に対応し、画像の最低空間周波数要素を表す。
RLE	ランレングス符号化（Run-length encoding）
ROI	対象とする範囲（Region of interest）
ROM	読出専用メモリ（Read Only Memory）
VLC	可変長符号（Variable-length code）

【０１２０】

概して、図６を参照すると、本開示は、エンコーダ１０１０及びそれに関連する操作方法に関連する。エンコーダ１０１０は、直接ＯＤｅｌｔａエンコーダとして実施されるという利点を持つ。さらに、本開示は、対応するデコーダ１０２０にも関連する。デコーダ１０２０は、逆ＯＤｅｌｔａデコーダとして実施されるという利点を持つ。該開示による実施形態では、有利には、上述した既知の差分符号化法のビット最適化バージョン、並びにその他データに関するレンジ最適化バージョンである直接ＯＤｅｌｔａ演算を利用する。ＯＤｅｌｔａ符号化が、可変長データワード、例えば８／１６／３２／６４ビットを利用し、かつ／又は元の値が１～６４ビットの範囲で表現される８／１６／３２／６４ビットのデータ要素の可変長符号化を利用するコンピュータハードウェア又は専用デジタルハードウェアにおいて利用され、対応する符号化値が１～６４ビットで生成される。勿論、エンコーダ１０１０及びデコーダ１０２０は、如何なる場合においても、データＤＡ１、例えば元データにどの種類の数値が含まれているのかを認識し、従って、その定義又は伝送が、ここでさらに明らかになることはない。数値範囲（ＭＩＮ及びＭＡＸ）が既知であること、並びにデータＤＡ１が利用され得ることが単に仮定される。

【０１２１】

既知の差分符号化法は、元（ＭＩＮ～ＭＡＸ）から結果（ＭＩＮ－ＭＡＸ～ＭＡＸ－Ｍ

ＩＮ）まで値の範囲を増加させる。この事は、該符号化法は、元データが正の値のみを含む場合、負の値もまた作成することを意味する。本開示によるＯＤｅｌｔａ演算は、対応する元の値の範囲にない値を作成することは決してなく、従って、使用されるデータ範囲を増加させることもなく、それ故に例えばエントロピー低減及び関連データ圧縮を実行する際に有利に利用される。例えば、既知の差分符号化法は、５ビット、即ち０～３１の値の範囲のデータストリームを用いて作動し、その結果、該差分符号化法によって生成されるデータ値は、－３１～＋３１の範囲、即ち６ビット（即ち符号ビット＋５ビット）を用いて実質的に表現され得る６３個の値になる。対照的に、直接ＯＤｅｌｔａ生成値は、上記の５ビットデータストリームから生成される際には、０～３１の範囲に依然としてある。さらに、既知の差分符号化法は、再起的に実施することは不可能であるが、本開示による直接又は逆ＯＤｅｌｔａ演算は、再帰的に実施可能であるにも関わらず、用いた値の範囲を依然として保存する。この値の範囲は、ビットに対して忠実である必要はなく、例えば、０～３１の値は５ビットで定義されるが、ＯＤｅｌｔａ演算は、任意の値の範囲、例えば０～２５の値の範囲を用いることができ、依然として適切に動作する。

【０１２２】

原則として、本明細書に記載されるＯＤｅｌｔａ法は、常に、既存のデータ範囲に基づいて直接機能することが可能であり、そのデータ範囲の例は以下に挙げられる。ＯＤｅｌｔａ法は、データにおいて生じる最低値（「ｌｏｗＶａｌｕｅ」）及びデータにおいて生じる最高値（「ｈｉｇｈＶａｌｕｅ」）を示す情報を伝達することによって強化することもできる。ｌｏｗＶａｌｕｅ≧ＭＩＮかつｈｉｇｈＶａｌｕｅ≦ＭＡＸであること、並びにこれらの値は任意選択のものであることに留意されたい。

【０１２３】

本開示による直接ＯＤｅｌｔａ演算及び逆ＯＤｅｌｔａ演算の２つの例が以下に記載されている。直接ＯＤｅｌｔａ演算及び逆ＯＤｅｌｔａ演算の最初の例は、例えば不揮発性（非一時的）機械可読データ記憶媒体に記録された１つ又は複数のソフトウェア製品を実行するように動作可能である電子ハードウェア及び／又はコンピュータハードウェアにおいて実施するのに、効率的であり、かつ比較的単純である。

【０１２４】

本開示による直接ＯＤｅｌｔａ演算又は逆ＯＤｅｌｔａ演算を実施する際には、有利には、元のデータ値列は全て正であり、最低値は０である。オプションで、何らかのオフセット値、即ちプリオフセット値又はポストオフセット値が、データ値のすべてが正となり、かつ最低値が「０」となるように、それらデータ値をシフトさせるのに利用され得る。本開示によるＯＤｅｌｔａ演算は、直接方式で全種類のデータと共に容易に利用され得る。本開示によるＯＤｅｌｔａ演算は、典型的には、データ圧縮を提供することが可能であり、即ち伝達データレートを減らすことができる。その理由は、オフセット値がすべての値に加算されるか、或はすべての値から引かれる際に、データ値の範囲は、より少ないビットで定義され得るからである。例えば、直接ＯＤｅｌｔａ演算又は逆ＯＤｅｌｔａ演算を適用する前の元データ値は、－１１～＋１８の範囲にあり、この範囲は、＋１１のオフセット値を用いて０～２９の範囲に変換することができ、その変換範囲はその後、５ビットで記述される。このようなプリオフセット値又はポストオフセット値が利用されない場合、元データ値は、それらを記述するのに少なくとも６ビットを必要とし、しばしば、実際には、完全な８ビット符号付バイトが便宜上利用される。

【０１２５】

データ範囲に対する同様の最適化が、一般化された直接ＯＤｅｌｔａ演算又は逆ＯＤｅｌｔａ演算を用いる際にもまた可能である。したがって、直接ＯＤｅｌｔａ演算若しくは逆ＯＤｅｌｔａ演算又は何らかのその他方法が、値の完全な範囲よりも小さいオフセット値で提示され得るデータ値を作成する場合、その範囲最適化は、ＯＤｅｌｔａ符号化法のどの段階においても実施され得る。オフセット値が、符号が負であろうと又は正であろうと用いられる際には、以下に図６、図７及び図８を参照して説明するとおり、オフセット値は、また、エンコーダ１０１０からデコーダ１０２０に伝送されなければならない。

【０１２６】
直接ＯＤｅｌｔａ演算は、例えばビット単位方式で元データＤＡ１を符号化するために、１ビット方式で容易に実施され得る。このような１ビット方式で、以下により詳細に説明されるとおり方法１及び方法３により、図６の元データＤＡ１においてビット値の変化がない場合には値「０」が作成され、元データＤＡ１においてビット値に変化が生じた場合には値「１」が作成される。元データの最初のビットに関する予測は、任意に値「０」であり、従って、元データＤＡ１における最初のビット値が保存される。或は、元データの最初のビット予測値を値「１」として利用することも任意に可能であるが、このような選択は、符号化において如何なる利益も与えることはない。そのため、その予測が常に１ビットデータに対してデフォルトで値「０」であると仮定される場合、如何なる選択も伝送される必要はない。即ち、所定の値「０」が、エンコーダ１０１０及びデコーダ１０２０によって利用されることにより、この予測を伝達する必要がなくなり、その結果データ圧縮の向上に繋がる。

【０１２７】
本開示による直接ＯＤｅｌｔａ符号化の例を以下に記載する。例示である元ビット列、即ち１７個の「１」と２０個の「０」とを含む３７個のビットが、以下のとおり式１で与えられる。

［式１］
０１０１０１１００１０００１０１０００００００００００１１１１１１１１１１１

【０１２８】
式１のエントロピーＥは、式２から計算可能である。
［式２］

$$E = 17 * \log_{10}\left(\frac{37}{17}\right) + 20 * \left(\frac{37}{20}\right) = 11.08523$$

【０１２９】
式２においてエントロピーを符号化するのに必要なビットの数、即ちMin_bitは、後掲の文書Ｐ７及びＰ８に記載されるとおり、つまり式３で与えられるとおり、シャノンの情報源符号化定理から計算可能である。
［式３］

$$\mathrm{Min\_bits} = \frac{\mathrm{E}}{\log_{10}(2)} = 36.82 \quad (\text{ビット})$$

【０１３０】
元ビット列が、上述のとおり直接ＯＤｅｌｔａ演算、即ち方法１及び方法３で処理されると、１３個の「１」と２４個の「０」が存在する、３７個のビットを含む以下のビット列が生成される。
［式４］
０１１１１１０１０１１００１１１１０００００００００００１０００００００００

【０１３１】

式４のエントロピーＥは、式５から計算可能である。
［式５］

$$E = 13 * \log_{10}\left(\frac{37}{13}\right) + 20 * \left(\frac{37}{24}\right) = 10.41713$$

【０１３２】
式５は、ビット最小値、即ち、式６によるMin_bitsで表現可能である。
［式６］

$$Min\_bits = \frac{E}{\log_{10}(2)} = 34.60 \quad (ビット)$$

【０１３３】
式４のビット列は、例えばランレングス符号化（ＲＬＥ）、ハフマン符号化、算術符号化、レンジ符号化、エントロピー変更符号化、又はＳＲＬＥ符号化のうちの少なくとも１つを用いた更なる符号化で処理され、データ圧縮が達成されるという利点を持つ。
【０１３４】
ＯＤｅｌｔａ演算は、その関連エントロピー符号化法が適用される場合、元データＤＡ１を表すのに必要なビット量を減らし、例えば、ＲＬＥ又はＳＲＬＥが、例えば式１にあるような元データの代わりに、例えば式４にあるような演算データに用いられ、この１ビット直接ＯＤｅｌｔａ演算は、即ち方法１及び方法３は、式１の元ビット列において多数の変化がある場合には複数の「１」を作成し、式１の元ビット列において相互に類似のビットの長いストリームがある場合には複数の「０」を生成する。
【０１３５】
ＯＤｅｌｔａ演算の逆バージョン、即ち方法１及び方法３の逆は、符号化データストリーム、即ちデータＤＡ２に値「１」がある場合には、ビット値を、値「０」から値「１」に、又は必要に応じて値「１」から値「０」に変化させ、データＤＡ２の符号化ストリームに「０」の値がある場合にはビット値を変化させることはない。ＯＤｃｌｔａ演算が、直接ＯＤｅｌｔａ演算が為されたデータＤＡ２のビットストリームに対して実行される場合、データＤＡ１の元ストリームは、復号データＤＡ５として再生される。しかしながら、上述のとおり、ＶＬＣ又はハフマン符号化等の更なる符号化が有利に利用されるが、この事も考慮される必要がある。これは、データＤＡ３がエントロピーエンコーダの正方向の演算を用いてデータＤＡ２から生成され、データＤＡ４がエントロピーデコーダの逆演算を用いてデータＤＡ３から生成されることを意味する。
【０１３６】
データＤＡ１の元ストリームは、符号化をそれに適用する前に、２以上のセクションに分割されるという利点を持つ。このような分割は、データＤＡ１の元ストリームを符号化する際により良い最適化が利用され得る機会を提供する。例えば、データＤＡ１における可変列が、直接ＯＤｅｌｔａ符号化される場合、即ち方法１及び方法３を利用して符号化される場合により多くの「１」を生成する。これに対して、フラットな不変列、即ち「フラット」列では、例えば後続のＶＲＬ符号化又はハフマン符号化にとって望ましい「０」が、より多く作成されるので、データＤＡ１を、上述のとおり別々に符号化できる複数のセクションに分割することによって、データＤＡ１を構成するビットストリーム全体に対

してエントロピーＥを低減することができる。したがって、このような分割が有利となる。

【０１３７】

本発明による直接ＯＤｅｌｔａ符号化の例を、相互に別々に符号化される複数のセクションが利用される場合について以下に説明する。元の単一ビット列を含む第１セクションは、以下式７のとおり、全体で１６個のビット、即ち７個の「１」及び９個の「０」を含む。

［式７］

０１０１０１１００１０００１０１

【０１３８】

ここで、Ｈ（Ｘ）＝４．７６２１かつＢ＝１５．８２であり、Ｈはエントロピーを示し、ＢはMax_bitを示す。式７の元ビット列が、直接ＯＤｅｌｔａ演算で処理される場合、対応変換ビット列が式８のように与えられる。

［式８］

０１１１１１０１０１１００１１１

【０１３９】

ここで、Ｈ（Ｘ）＝４．３１５８かつＢ＝１４．３４である。

【０１４０】

元の単一ビット列を含む第２セクションは、以下式９に示すとおりのビットを含む。

［式９］

０００００００００００１１１１１１１１１１

【０１４１】

ここで、Ｈ（Ｘ）＝６．３１１３かつＢ＝２０．９７である。式９の元ビット列が直接ＯＤｅｌｔａ演算で処理されると、対応変換ビット列は、式１０のとおり与えられる。

［式１０］

０００００００００００１０００００００００

【０１４２】

ここで、Ｈ（Ｘ）＝１．７４６０かつＢ＝５．８０である。これらの例では、上述したとおり、Ｈ（Ｘ）はエントロピーＥを表し、Ｂは、符号化に要するビット最小数を表す。

【０１４３】

この例における式７及び式１０による最良の圧縮は、両セクションが別々に直接ＯＤｅｌｔａ演算で処理される場合に達成される（即ち、１４．３４ビット＋５．８０ビット＝全体で２０．１４ビットに符号化）。これは、もともと必要とされていた３６．８２ビットよりも少ないビットを要し、即ち直接ＯＤｅｌｔａ演算ビットは３４．６０ビット、又は分割後に要したビット数（－１５．８２ビット＋２０．９７ビット－３６．７９ビット）を要する。有利には、データＤＡ１の元ビットストリームのセクションへの分割を、元データＤＡ１、及び変更データ（即ちデータＤＡ２に含まれるような変更データ）の対応エントロピーＨを１つずつ分析することによって自動的に実行する。

【０１４４】

データＤＡ１に複数の長いランセクションがある場合に、ビット値が配列に沿って急激に変化する十分に大きなデータ領域があることを前提として、オプションで、データＤＡ１の部分を符号化される新規セクションに単に分割することによって粗い方式で、データ圧縮が実施される。オプションで、データＤＡ１の幾つかのセクションは、例えば個々の異なるビットが比較的少ない、相互に類似のビットの長いランが存在する場合、直接ＯＤｅｌｔａ演算を利用することなく符号化される。このような場合、直接ＯＤｅｌｔａ演算は、データ圧縮目的には有意な利益をもたらさない。

【０１４５】

データＤＡ１をより小さいセクションに分割することは、符号化データＤＡ２にデータを付与する更なるオーバーヘッドを生成するという欠点を有する。このようなオーバーヘッドは、例えば、全ての新しいセクションに関連するデータビット量又はデータバイト量

を示す情報を含む。しかしながら、少なくとも特定量のオーバーヘッドデータ値を伝達する必要があると常に認められ、従って、所与のデータが２つのデータセクションに分割された場合には、追加のオーバーヘッドデータ値が１つのみ存在することになる。

【０１４６】

後に復号され得る符号化ビットストリームを達成するためには、エントロピー符号化が、直接ＯＤｅｌｔａ演算、例えばＶＬＣ、ハフマン符号化、算術符号化、レンジ符号化、ＲＬＥ、ＳＲＬＥ、ＥＭ等の後に有利に実施される。実際のデータ符号化と比較して、算出エントロピーＥ及び最小ビット見積値に基づいて最適化計算を実行する方がより簡単であり、かつ計算上より効率的である。このような順番で実行することにより、顕著な速度の最適化が可能となり、しばしば、符号化データＤＡ２において最適なデータ圧縮の結果が達成される。或は、元のビット、アルファベット、数字、バイト及びワードのデータ、即ちデータＤＡ１にあるものが、何らかの他の方法で最初に符号化されることにより、エントロピー最適化ビットストリームが生成され、その後、直接ＯＤｅｌｔａ演算を用いてエントロピー最適化ビットストリームを変更し、対応符号化データ、即ちデータＤＡ２を提供するように、エントロピー最適化を実行することも可能である。さらに、このＯＤｅｌｔａ演算データは、データＤＡ２から、さらに他の符号化方法を用いて符号化し、データＤＡ３を生成することもできる。

【０１４７】

一般化された直接ＯＤｅｌｔａ演算は、データＤＡ１において用いられる様々な値を記述するパラメータ、即ち、その様々な値を提示するのに必要とされるビットの値又は数を利用する。さらに、ＯＤｅｌｔａ演算は、正及び負のオフセット値、言い換えると正及び負の「ペデスタル（基礎）」値の使用を可能にする方法において利用される。例えば、データＤＡ１が７個のビットを用いて提示され場合であって、即ち使用可能な「０」～「１２７」の値を有するが、「６０」～「１１５」の範囲の値のみを含む場合において、－６０のオフセット値がデータＤＡ１に適用されると、これによって、６個のビットのみを含む値として表すこともできる、「０」～「５５」の範囲の値を有する変換データが生成され、つまりある程度のデータ圧縮が、これによって達成可能となる。従って、この一般化直接ＯＤｅｌｔａ演算は、全範囲のデータ値が、即ち７個のビットにおいて表され、かつ８ビットバイトによって従来表されるデータ値がデータＤＡ１に存在する場合に、結果を向上させる。

【０１４８】

本開示によれば、直接ＯＤｅｌｔａ値、即ち方法１は、正の値（ｌｏｗＶａｌｕｅ＝ＭＩＮ＝０かつｈｉｇｈＶａｌｕｅ＝ＭＡＸ＝１２７、ｗｒａｐＶａｌｕｅ＝１２７－０＋１＝１２８）のみを有するデータに関する、以下の例示ソフトウェアコードの抜粋によって記述される手順を用いて容易に計算可能である。

```
wrapValue = power(2, bits) = power(2, 7) = 128
prediction Value（予測値） = (lowValue + highValue + 1) div 2 = (wrapValue + 1)
div 2 + lowValue = 64
for all pixels（全てのピクセルについて）

begin
    if(originalValue >= predictionValue) then
        ODeltaValue = originalValue - predictionValue
    else
        ODeltaValue = wrapValue + originalValue - predictionValue
    predictionValue = originalValue
End
```

【０１４９】

さらに上記ＯＤｅｌｔａ演算を説明するために一例を以下に提供する。値の元配列は、

式１１のとおりである。
［式１１］
65, 80, 126, 1, 62, 45, 89, 54, 66
【０１５０】
対応する従来の差分符号化値は式１２のとおりである。
［式１２］
65, 15, 46, -125, 61, -17, 44, -35, 12
【０１５１】
対応直接ＯＤｅｌｔａ符号化値は式１３のとおりである。
［式１３］
1, 15, 46, 3, 61, 111, 44, 93, 12
【０１５２】
ここで、パラメータwrapValue内のラップアラウンドが利用される。
【０１５３】
逆ＯＤｅｌｔａ演算、即ち方法１が、逆ＯＤｅｌｔａ値を生成するために使用可能であり、例えば以下の例示ソフトウェアコードによって実施される。

```
wrapValue = power(2, bits) = power(2, 7) = 128
predictionValue（予測値） = (wrapValue + 1) div 2 + lowValue = 64
for all pixels（全てのピクセルについて）

begin
    ODeltaValue = originalValue + predictionValue
    if (ODeltaValue >= wrapValue) then
        ODeltaValue = ODeltaValue - wrapValue
    predictionValue = ODeltaValue
end
```

【０１５４】
このソフトウェアコードが、式１３に対して実行、適用された場合、式１４で与えられる値が生成される。
［式１４］
65, 80, 126, 1, 62, 45, 89, 54, 66
【０１５５】
本例では、ｗｒａｐＶａｌｕｅが２の冪乗の値として用いられる。これは必須ではなく、負の値がまた利用可能であるか、所与のデータ列においてプリオフセットによって範囲が変更される場合、ｗｒａｐＶａｌｕｅが、データの最高値より大きい任意の値、又は用いられる範囲よりも大きい値であってもよい。この特徴を示す更なる例を以下に示す。
【０１５６】
図６を参照した上述の説明を要約すると、本開示は、エンコーダ１０１０及びデコーダ１０２０に関連するものである。オプションで、エンコーダ１０１０及びデコーダ１０２０は、３０によって全体が示されているコーデック装置と組合せて利用される。エンコーダ１０１０は、元入力データＤＡ１を受信するように動作可能であり、元入力データＤＡ１は、例えば直接ＯＤｅｌｔａ法を用いて符号化されることにより、対応符号化データＤＡ２又はＤＡ３が生成される。符号化データＤＡ２又はＤＡ３は、オプションで、通信ネットワーク１０４０を介して伝達されるか、又はデータ記憶媒体１０５０、例えば光ディスク読取専用メモリ（ＲＯＭ）等のデータ媒体に記憶される。デコーダ１０２０は、例えば通信ネットワーク１０４０を介してストリーム配信される符号化データＤＡ２又はＤＡ３、又はデータ記憶媒体１０５０に提供される符号化データＤＡ２又はＤＡ３を受信し、かつ逆の方法、例えば逆ＯＤｅｌｔａ法を適用することにより、例えば元データＤＡ１に実質的に類似する対応復号データＤＡ５を生成するように動作可能である。エンコーダ１

０１０及びデコーダ１０２０は、例えば本明細書における例示実施形態として提供されるコードのように、１つ又は複数のソフトウェア製品を実行するように動作可能である、デジタルハードウェア、例えばコンピュータハードウェアを用いて実施されるという利点を持つ。或は、エンコーダ１０１０及び／又はデコーダ１０２０は、専用デジタルハードウェアを用いて実施される。

【０１５７】

エンコーダ１０１０において実行されるＯＤｅｌｔａ法は、図７に記載される工程を利用するものである。任意の第１工程１１００において、入力データＤＡ１が処理され、そのデータ要素の値の範囲が見出される。任意の第２工程１１１０において、データ要素を正のレジームに変換することにより変換要素の対応セットを生成するために、その値の範囲から、オフセット、即ちプリオフセットが計算される。第３工程１１２０においては、第２工程１１１０において任意に変換された要素が、次いで、直接ＯＤｅｌｔａ符号化を受けることにより、対応ＯＤｅｌｔａ符号化値が生成される。第４工程１１３０においては、ＯＤｅｌｔａ符号化値、並びに任意のオフセット値、最小値（ｌｏｗＶａｌｕｅ）、及び／又は最大値（ｈｉｇｈＶａｌｕｅ）が、次いで、例えばランレングス符号化（ＲＬＥ）、レンジ符号化、又はハフマン符号化を用いて別々に符号化され、データＤＡ２からデータＤＡ３が生成される。オフセット値、最小値（ｌｏｗＶａｌｕｅ）、及び／又は最大値（ｈｉｇｈＶａｌｕｅ）は、常に圧縮可能であるという訳ではなく、従って、それらはエンコーダ１０１０からデコーダ１０２０に適切なビット量を用いて伝達されることを要する。さらに、オフセット値、最小値（ｌｏｗＶａｌｕｅ）、及び／又は最大値（ｈｉｇｈＶａｌｕｅ）は、直接ＯＤｅｌｔａ演算に関する任意の特徴であり、例えば、オフセット値は、特定の状況では、値「０」を有し、ｌｏｗＶａｌｕｅは、値ＭＩＮを有し、ｈｉｇｈＶａｌｕｅは、値ＭＡＸを有する。即ち、変換は全く適用されずに、全範囲が用いられる。特に、直接ＯＤｅｌｔａ演算が、１ビットデータ、即ちビット単位の符号化のために実施される場合、オフセット値を必要とすることは全くなく、そのため、工程１１００及び１１０は常に無視される。オフセット値が、工程１１１０においてもまた用いられる場合には、最高値と最低値を提示する範囲値は、その中で更新されなければならない。異なる値の数、即ちｗｒａｐＶａｌｕｅは、デコーダ１０２０によってまた識別されなければならず、或は、さもなければエンコーダ１０１０は、圧縮データ内においてそれをデコーダ１０２０に伝達しなければならない。オプションで、デフォルトｗｒａｐＶａｌｕｅ（＝ｈｉｇｈＶａｌｕｅ－ｌｏｗＶａｌｕｅ＋１）が、エンコーダ及びデコーダにおいて用いられる。オプションで、エンコーダ１０１０及びデコーダ１０２０の少なくとも１つが、例えば入力データＤＡ１を符号化のためのセクションに分割してデータＤＡ１の最適な圧縮を提供し、符号化データＤＡ２を生成する最適な方式を見出すために、再帰的な方式で動作する。

【０１５８】

デコーダ１０２０において実行される逆ＯＤｅｌｔａ法は、図８に記載される工程を利用する。第１工程１２００において、データＤＡ２／ＤＡ３又はＤＡ４は、上述の工程１１３０において利用されるものとは逆の符号化を受けることにより、復号ＯＤｅｌｔａデータが生成され、復号ＯＤｅｌｔａデータは、ＯＤｅｌｔａ符号化値を有し、かつ任意の別のオフセット値を有する。第２工程１２１０においては、ＯＤｅｌｔａ符号化値は、復号され、データ要素列が生成される。第３工程１２２０においては、データ要素列が、最適プリオフセット値を用いて変換され、復号データＤＡ５が生成される。特定の状況においては、このような変換は値「０」に設定され、つまり、有効な変換が適用されることはない。この場合も、例えば１ビット符号化、即ちビット単位の符号化を実行する場合、オフセット値を利用する必要なく、上記方法を実行することが可能であり、それによって、工程１２２０を無視することが可能になる。さらに、デコーダ１０２０は、受信したデータ要素を適切な方式で復号することを可能にするためにｗｒａｐＶａｌｕｅを識別することも要する。

【０１５９】

上記オフセットを利用し、正の値のみを取得することによって、データＤＡ２又はＤＡ３におけるより効率的なデータ圧縮が達成可能となる。すべてのデータ値が、既に正の値である場合、如何なるオフセット値を追加する必要はない。勿論、以下の例に示すとおり、負のオフセット値は、利用可能な範囲を小さくするために任意に利用されるが、必須ではない。

【０１６０】

図７及び図８の方法は、オプションで、ＯＤｅｌｔａ符号化を受ける利用可能な値のみを用いることによって更に最適化され得る。このような最適化には、用いる値が既知であることが必要とされる。例えば、上述の例においては、１（＝元の最小値）～１２６（＝元の最大値）の値のみが元データセットＤＡ１に存在する。このため、オフセット値は、１である（->ｌｏｗＶａｌｕｅ＝元の最小値－オフセット＝１－１＝０かつｈｉｇｈＶａｌｕｅ＝元の最大値－オフセット＝１２６－１＝１２５）。プリオフセット値が元データＤＡ１から引かれた場合、結果として式１５にある次の値が得られる。

［式１５］

64, 79, 125, 0, 61, 44, 88, 53, 65

【０１６１】

式１５から、最大値として１２５が決定され（ｈｉｇｈＶａｌｕｅ＝元の最大値（ｏｒｉｇｉｎａｌ ｍａｘ）－オフセット＝１２６－１＝１２５）、その結果、「数字」（＝最大差分値（ｍａｘｉｍｕｍ Ｄｅｌｔａ ｖａｌｕｅ）＝ｈｉｇｈＶａｌｕｅ － ｌｏｗＶａｌｕｅ）は、そうすると、１２５となり、即ちｗｒａｐＶａｌｕｅは、最小で１２６となり得る（＝数字＋１＝ｈｉｇｈＶａｌｕｅ－ｌｏｗＶａｌｕｅ＋１）。すると、これらの値を保存するか、及び／又は伝達する必要があるため、前の例は、以下のとおりプロセス値を変化させることによって変更することができる。

wrapValue = 126　　（「０」～「１２５」=>１２６個の異なる値）
prediction Value（予測値）= (highValue + lowValue + 1) div 2 = (wrapValue + 1) div 2 + lowValue = 63

【０１６２】

対応する直接ＯＤｅｌｔａ演算の値は式１６で与えられている。

［式１６］

1, 15, 46, 1, 61, 109, 44, 91, 12

【０１６３】

全ての「負の差分値」が、今や、２分の１に減じられている（即ち＝範囲変化＝１２８－１２６）ことが理解される。同様に、デコーダ１０２０においても、プロセス値は以下のとおり変化させなければならない。

wrapValue = 126
predictionValue = (wrapValue + 1) div 2 + lowValue = 63

【０１６４】

対応する逆のＯＤｅｌｔａ値は、以下の式１７のとおりである。

［式１７］

64, 79, 125, 0, 61, 44, 88, 53, 65

【０１６５】

プリオフセット値を式１７に加算すると、式１５における元データに対応して、以下の式１８の結果が得られる。

［式１８］

65, 80, 126, 1, 62, 45, 89, 54, 66

【０１６６】

本例では、値の範囲はほぼ全体であり、従って、オフセット値及び最大値（ｈｉｇｈＶａｌｕｅ）を用いて直接ＯＤｅｌｔａ演算を適用することにより得られる利益は比較的そ

れほど大きいものではない。しかしながら、それら値が適切に伝達される場合には、エントロピーＥの低減が依然として達成可能であり、即ち、頻度テーブル又は符号テーブルにおいて値の数を減らすことができる。この範囲がより小さい場合に最大の利益が得られる。

【０１６７】

データを符号化及び復号する実践的な１ビットの直接及び逆ＯＤｅｌｔａ法の例示実施形態、即ち、方法１又は方法３が、実行可能なコンピュータソフトウェアコードにより以下に提供される。これらの方法は、上記の直接及び逆ＯＤｅｌｔａ演算、即ち方法１又は方法３を利用する。ソフトウェアコードは、コンピュータハードウェアで実行される際に動作可能であり、１つのバイトバッファから別のバイトバッファに対してビットを処理する。ソフトウェアコードでは、ＧｅｔＢｉｔ、ＳｅｔＢｉｔ、及びＣｌｅａｒＢｉｔの関数が、常に、ＨｅａｄｅｒＢｉｔｓ値を更新する。ＨｅａｄｅｒＩｎｄｅｘ値もまた、次のビットが次のバイトにある場合に更新される。オプションで、ソフトウェアコードは、１セットのＨｅａｄｅｒＩｎｄｅｘ値及びＨｅａｄｅｒＢｉｔｓ値のみがソース及び宛先に関し用いられるように最適化することができ、その結果、所与のビットが宛先バッファに書き込まれる場合のみ値が更新される。

```
procedure EncodeODelta1u(APtrSrc : PByte; ASrcDstBitLen : PCardinal; APtrDst : PByte)
var
 iSrcHeaderIndex, iSrcHeaderBits, iIndex,
 iDstHeaderIndex, iDstHeaderBits : Cardinal;
 bBit, bLastBit : Boolean;
begin
 // オフセットをリセット。
 iSrcHeaderIndex := 0;
 iSrcHeaderBits := 0;
 iDstHeaderIndex := 0;
 iDstHeaderBits := 0;

 // 差分値を初期化。
 bLastBit := False;

 // 全てのビットを処理。
 for iIndex := 0 to ASrcDstBitLen^-1 do
 begin
  // ビットを読み取る。
  bBit := GetBit(APtrSrc, @iSrcHeaderIndex, @iSrcHeaderBits);

  // 現ソースビットが前のソースビットと異なる場合には宛先ビットをセット。
  if (bBit <> bLastBit) then
  begin
   SetBit(APtrDst, @iDstHeaderIndex, @iDstHeaderBits);
   bLastBit := bBit;
  end
  else ClearBit(APtrDst, @iDstHeaderIndex, @iDstHeaderBits);
 end;
end;
```

```
function DecodeODelta1u(APtrSrc : PByte; ASrcDstBitLen : PCardinal; APtrDst : PByte) :
Boolean;
var
    iSrcHeaderIndex, iSrcHeaderBits, iIndex,
    iDstHeaderIndex, iDstHeaderBits : Cardinal;
    bBit, bLastBit : Boolean;
  begin
    // オフセットをリセット。
    iSrcHeaderIndex := 0;
    iSrcHeaderBits := 0;
    iDstHeaderIndex := 0;
    iDstHeaderBits := 0;

    // 差分値を初期化。
    bLastBit := False;

    // 全てのビットを処理。
    for iIndex := 0 to ASrcDstBitLen^-1 do
    begin
     // ビットを読み取る。
     bBit := GetBit(APtrSrc, @iSrcHeaderIndex, @iSrcHeaderBits);

     // ソースビットが真(True)である場合にはビット値を変更。
     if (bBit = True) then
     begin
      if (bLastBit = True) then
       bLastBit := False
      else bLastBit := True;
     end;

     // ビット値(真(True)又は偽(False))に基づいて宛先ビットをセット。
     if (bLastBit) then
      SetBit(APtrDst, @iDstHeaderIndex, @iDstHeaderBits)
     else ClearBit(APtrDst, @iDstHeaderIndex, @iDstHeaderBits);
    end;
  end;
```

【０１６８】

上述の直接及び逆ＯＤｅｌｔａ演算、即ち方法１又は方法３は、例えばビデオデータ、画像データ、音声データ、グラフィックデータ、地震データ、医療データ、測定値、参照

数字及びマスク等のデジタルフォーマットである任意の種類のデータを圧縮するのに有利に利用される。さらに、１つ又は複数のアナログ信号も、対応デジタルデータに最初に変換させる場合に、例えば圧縮前にＡＤＣを用いることによって、直接ＯＤｅｌｔａ演算を用いて圧縮可能である。逆ＯＤｅｌｔａ演算を用いる際、データを変換して１つ又は複数のアナログ信号に戻すことが望まれる場合には、演算の後にＤＡＣを用いることができる。しかしながら、直接ＯＤｅｌｔａ演算それ自体は、データを圧縮するのに通常は効果的ではないものの、例えば可変長符号化（ＶＬＣ）、算術符号化、レンジ符号化、ランレングス符号化、ＳＲＬＥ、エントロピー変更等の他の符号化法と組合せて利用される場合には効果的なデータ圧縮を提供できることが理解される。これらの符号化法は、直接ＯＤｅｌｔａ演算がエンコーダ１０１０において利用された後に、データＤＡ２に対して用いられる。生じるデータがデコーダ１０２０において実施される逆ＯＤｅｌｔａ演算に伝達される前に、符号化データＤＡ２は、対応的に復号されて元に戻されなければならない。ＯＤｅｌｔａ演算は、他の種類のエントロピー変更演算と共に利用することもできる。特定の状況においては、直接ＯＤｅｌｔａ演算は、エントロピーＥの増加を生じさせることもあり、データ圧縮アルゴリズムは、データの符号化に用いるための直接ＯＤｅｌｔａ演算を、それが有利なデータ圧縮性能を提供する場合のみ、選択的に利用するように有利に動作可能であり、例えば、それは圧縮されるデータの性質に基づいて選択的に利用されたり、例えば上述のとおり入力データＤＡ１の選択部分に選択的に利用されたりする。

【０１６９】

直接ＯＤｅｌｔａ演算は、例えば、本願において援用される米国特許出願ＵＳ１３／５８４，００５に記載されるようなブロックエンコーダと組合せて利用されること等が考案されている。逆ＯＤｅｌｔａ演算は、本願において援用される米国特許出願ＵＳ１３／５８４，０４７に記載されるようなブロックデコーダと組合せて利用されることが考案されている。オプションで、直接ＯＤｅｌｔａ演算及び逆ＯＤｅｌｔａ演算は、本願において援用される米国特許出願ＵＳ１３／６５７，３８２に記載されるようなマルチレベル符号化法と組合せて利用されるという利点を持つ。２進状態を含む全ての種類の１ビットデータ、例えばデータＤＡ１に存在するもの等を１ビットバージョンの直接ＯＤｅｌｔａ演算で処理されることにより対応変換データを生成し、対応変換データは、次いで、その後、実際のエントロピー符号化を受けることにより、符号化データＤＡ２又はＤＡ３が生成されるという利点を持つ。オプションで、上述のとおり、直接ＯＤｅｌｔａ演算が、元データＤＡ１の性質に応じて選択的に利用される。

【０１７０】

オプションで、直接ＯＤｅｌｔａ演算の前又は後に、データのエントロピーを変更する他の方法を利用することができる。例えば、一般化されたバージョンの直接ＯＤｅｌｔａ演算の中でマルチビットデータに対して直接、直接ＯＤｃｌｔａ演算を用いることもできる。さらに、用いられるビット全てをビットの連続配列に最初に入力した後に、上記１ビットバージョンの直接ＯＤｅｌｔａ演算をマルチビットデータに対して有利に利用する。

【０１７１】

エンコーダ１０１０において直接ＯＤｅｌｔａ演算と組合せて複数の方法がデータ圧縮のために利用される場合、対応する逆演算が、例えば以下のとおり、デコーダ１０２０において逆順に実行される。

【０１７２】

以下の処理がエンコーダ１０１０において利用される。

［式１９］

［データＤＡ１］　　＝＞直接ＯＤｅｌｔａ（方法２）
＝＞ＶＬＣ－
＝＞ＥＭ
＝＞算術符号化
＝＞　［データＤＡ３］

【０１７３】

以下の方法の逆順がデコーダ１０２０において利用される。
［式２０］
［データＤＡ３］　　＝＞逆算術符号化
＝＞逆ＥＭ
＝＞逆ＶＬＣ
＝＞逆ＯＤｅｌｔａ（方法２）
＝＞［データ ＤＡ５］
【０１７４】
ここで、「ＶＬＣ」は可変長符号化を示し、「ＥＭ」はエントロピー変更を示す。
【０１７５】
上記に記載したＯＤｅｌｔａ演算は、可逆的であり、かつ無損失である。さらに、上記ＯＤｅｌｔａ演算は、オプションで、例えばビット単位の符号化を実行する際には１ビットデータストリームに対して、さらにそれだけではなくその他データに対しても、特に実施することが容易に可能である。すべての種類のデータが、一般化バージョンの直接ＯＤｅｌｔａ演算を用いて容易に処理され得るという利点を持つ。そして、直接ＯＤｅｌｔａ演算はデータが圧縮される際に利用され、対応する逆ＯＤｅｌｔａ演算は圧縮データを展開する際に利用されるという利点を持つ。オプションで、ＯＤｅｌｔａ演算が利用される場合、直接ＯＤｅｌｔａ演算と、それに対応する逆演算が、逆順に利用され、言い換えると、逆ＯＤｅｌｔａ演算は、元ビットストリームに対して最初に一時的に実行され、その後直接ＯＤｅｌｔａ演算が続き、元ビットストリームが再生される。一方のＯＤｅｌｔａ演算はエントロピーを増やし、他方のＯＤｅｌｔａ演算はエントロピーを減らす。直接ＯＤｅｌｔａ演算がエントロピーを全く変更してはならず、次いで逆ＯＤｅｌｔａ演算もエントロピーを変更しないというのは極めて稀有なケースである。例えば方法１のために直接ＯＤｅｌｔａ演算及び逆ＯＤｅｌｔａ演算が用いられる場合、これらの演算の逆順は、方法４の通常の順序と同様である。順序の同様の変更が、方法２及び方法３についても可能である。
【０１７６】
１ビットバージョンにおいて、即ちビット単位方式でデータを符号化するために、直接ＯＤｅｌｔａ演算は、予測無しで有利に開始し、即ち、デフォルトで初期値「０」の予測を仮定する。一般化バージョンにおいては、ＯＤｅｌｔａ演算は、使用可能なデータ範囲の半分を表す予測を用いて開始し、例えば、５ビットがデータＤＡ１の入力データ値に用いられる場合、即ち「０」～「３１」の範囲の３２個の異なる値が用いられる場合、予測値は３２/２＝１６である。有利には、ＯＤｅｌｔａ演算には、データ要素が該演算を用いて処理されるために、使用可能なデータ範囲に関する情報が提供される必要がある。
【０１７７】
上記に記載される開示の実施形態によれば、データＤＡ１においてビット又は任意のデジタル値として提示されるエントロピーＥを低減することが可能になる。直接ＯＤｅｌｔａ演算によって、ほぼ常に、差分符号化と比較してより向上したエントロピー低減を行える。差分符号化がバイトラップアラウンドと共に用いられ、元予測を用いた差異ＯＤｅｌｔａ演算（方法１）が、ｗｒａｐＶａｌｕｅ＝２５６、ｌｏｗＶａｌｕｅ＝ＭＩＮ＝０、及びｈｉｇｈＶａｌｕｅ＝ＭＡＸ＝２５５の値を用いる場合のみ、その中で同一の出力結果を生成する。別の直接ＯＤｅｌｔａ法が用いられる場合、又は入力データにおいて全データ範囲が利用可能ではない場合、ＯＤｅｌｔａ演算は、選択した方法、又はｌｏｗＶａｌｕｅ及び／若しくはｈｉｇｈＶａｌｕｅを送信することによってより良い結果を生成するが、即ち、これら送信されるものはｗｒａｐＶａｌｕｅもまた自動的に変更する。エントロピーが小さいほど、より高いデータ圧縮比でデータを圧縮することが可能になる。より高いデータ圧縮比は、より小さい容量のデータ記憶媒体の利用を可能にし、また、圧縮データを伝送する場合に、より遅いデータ帯域幅の利用も可能にし、対応するエネルギー消費の低減も可能にする。
【０１７８】

上記において、差分と合算の方式の計算がエンコーダ１０１０において実行され、かつ対応する逆の計算がデコーダ１０２０において実行されることが理解される。エンコーダ１０１０において用いられる他の予測方法を用いることも可能であり、対応する逆予測が次いでデコーダ１０２０において実行される。この事は、実際、少なくとも４つの異なる直接ＯＤｅｌｔａ方法、並びに少なくとも４つの対応する逆ＯＤｅｌｔａ方法が存在することを意味している。これらの方法の詳細かつ厳密な説明は以下のとおりである。オプションで、上記計算は、再帰的な方式で実行されることにより、符号化データＤＡ２（又はＤＡ３）において高度なデータ圧縮が得られる。このような帰納的な計算を実行する際には、変化する数の範囲が、再帰的計算が何回利用されてきたかに応じて利用される。例えば、エンコーダ１０１０においては、以下の順序の計算がデータＤＡ１に対して実行され、符号化データＤＡ２（又はＤＡ３）が生成される。

【０１７９】

［式２１］

［データＤＡ１］　　ｅ直接ＯＤｅｌｔａ（方法３）＝＞
ｅ直接ＯＤｅｌｔａ（方法３）＝＞
ｅＥＭ＝＞
ｅ直接ＯＤｅｌｔａ（方法１）＝＞
ｅＶＬＣ　［データＤＡ３］

【０１８０】

デコーダ１０２０において対応する逆演算が実行される。

【０１８１】

［式２２］

［データＤＡ３］　　ｄＶＬＣ＝＞
ｄ逆ＯＤｅｌｔａ（方法１）＝＞
ｄＥＭ＝＞
ｄ逆ＯＤｅｌｔａ（方法３）＝＞
ｄ逆ＯＤｅｌｔａ（方法３）　［データＤＡ５］

【０１８２】

式２１（方法１に対応）、式２２（方法２に対応）、式２３（方法３に対応）、及び式２４（方法４に対応）によって示されるとおり、データがこれら４つの方法において演算処理されるたびに、任意にすべての方法を用いることを試すことが可能である。その理由は、これらの方法のうち１つが、その他の方法よりも、処理されるデータのエントロピーを減らせる可能性があるからである。エンコーダ１０１０及び／又はデコーダ１０２０内における方法の使用を最適化する上で、選択される１つの方法又は複数の方法が、必要とされるデータにおける情報量と比較してエントロピーを減少させる限り、何度も同一又は異なる方法を用いることは有利である。従って、方法１～４は、数値を符号化するために使用可能であるが、「数値」とは、その定義において、ビット単位方式の符号化ビットストリームにあるような１ビットデータ並びに非２進数、並びにマルチビット値を包含する。

【０１８３】

差分演算は、連続数値の余りを表し、それに対応して合算演算は、連続数値の合計を表す。エンコーダ１０１０において実行されるこれらの演算には、デコーダ１０２０におけるそれぞれ独自の対応する逆演算がある。差分又は合計は、現入力値、及び予測値として用いられる前の入力値又は結果値に基づいて計算され得る。他の予測値も用いることができ、それらは、例えば、デコーダにおいて逆を実行することが可能である限りにおいて、エンコーダにおいて以前の入力値及び出力値を用いることにより予測を作成し得る。

【０１８４】

このような方法の何れも、エンコーダ１０１０及びデコーダ１０２０内で有意にデータを圧縮するものではないが、すべての方法は、エントロピーを低減するのに有利に利用され、その結果、他の圧縮法により、次いで、エントロピー低減データをより効率的に圧縮

することができる。このような他の圧縮法は、オプションで、ハフマン符号化、算術符号化、レンジ符号化、ＲＬＥ符号化、ＳＲＬＥ符号化、エントロピー変更符号化の少なくとも１つである。しかしながら、全ての方法に関し、例えばデータの可逆圧縮及びそれに続く可逆展開が達成し得る場合には、幾つかの数値を伝送する必要があり、その幾つかの数値を用いて上記演算とその逆演算とが常に正確に実施し得る。勿論、エンコーダ１０１０及びデコーダ１０２０は、どのような種類の数値が入力データＤＡ１に含まれているかに関する情報を有する。有利には、数値範囲、即ちＭＩＮ及びＭＡＸによって定義される数値範囲が認識されることが想定されている。原則、方法は、常に、直接既存のデータ範囲に基づいて機能し得る。上記演算が必要とする数値は、生じる最低値（ｌｏｗＶａｌｕｅ）及び生じる最高値（ｈｉｇｈＶａｌｕｅ）であり、ｌｏｗＶａｌｕｅはＭＩＮ以上であり、ｈｉｇｈＶａｌｕｅはＭＡＸ以下である。

【０１８５】

これらの値に基づいて、他の必要な数値を導くことができる。これらの値は、様々な形態で伝送され、欠損値が有利に計算されるという利点を持つ。例えば、セット［「ｌｏｗＶａｌｕｅ」、「ｈｉｇｈＶａｌｕｅ」、「ｎｕｍｂｅｒ（数）」］からの２つの値が既知である場合、その「ｎｕｍｂｅｒ（数）」は、［ｈｉｇｈＶａｌｕｅ －ｌｏｗＶａｌｕｅ］であり、次いで、第３の値は、それらから計算することができる。データＤＡ２において特定の値を省略し、次いでデコーダ１０２０においてそれら値を導くことは、データＤＡ２においてより大きな程度のデータ圧縮を提供することを可能にする。

【０１８６】

これらの値に加えて、第１の値、即ち「ｐｒｅｄｉｃｔｉｏｎ」（「予測」）の計算において前の値として用いられ得る数Ｐが必要とされる。「０」と「ｎｕｍｂｅｒ」（「数」）の値との間の値は、常に、数Ｐ、即ち「ｐｒｅｄｉｃｔｉｏｎ」（「予測」）に関して選択され得る。さらに、上記演算が、エンコーダ２０においてデータＤＡ２／ＤＡ３又はＤＡ４を復号する際に回復可能に機能するためには、つまり、演算が生成する値の範囲をできる限り小さく縮小させるためには、値「ｗｒａｐＶａｌｕｅ」が上記演算に提供される必要がある。しかしながら、この「ｗｒａｐＶａｌｕｅ」は、「ｎｕｍｂｅｒ」（「数」）よりも大きくなければならず、有利には、それは値「ｎｕｍｂｅｒ」（「数」）＋１を有する。オプションで、例えばデータＤＡ１が大きい値よりも小さい値をより多く含むと仮定した場合、データＤＡ１の性質に応じて、第１「ｐｒｅｄｉｃｔｉｏｎ」（「予測」）値には、上記のとおり「０」が選択され得る。或は、データＤＡ１が小さい値よりも大きい値をより多く含むと仮定した場合、第１「ｐｒｅｄｉｃｔｉｏｎ」「予測」値には、「ｎｕｍｂｅｒ」「数」と等しい値が選択され得る。仮定が値の大きさに関して為されていない場合、「ｐｒｅｄｉｃｔｉｏｎ」（「予測」）値には値「（ｗｒａｐＶａｌｕｃ＋１）÷２＋ｌｏｗＶａｌｕｃ」を用いることが望ましい。

【０１８７】

本開示の実施形態を実施する際にコンピュータハードウェアにおいて実行する演算の例について以下に説明する。

【０１８８】

エンコーダ１０１０において、第１直接差分演算、即ち方法１は、有利には以下のとおり実施される。全てのデータ値に関し、入力値、即ち「ｏｒｉｇｉｎａｌ」値（「元」）値に対応する出力値、即ち「ｒｅｓｕｌｔ」（「結果」）がソフトウェアループにおいて計算される。

```
result = original - prediction
if result < lowValue then result = result + wrapValue
```

【０１８９】

最終的に、次の入力に関する予測値が現入力と等しい値に設定され、即ち、以下とされる。

Prediction = original

【０１９０】

デコーダ１０２０において、第１逆差分演算、即ち、方法１が、有利には、以下のとおり実施される。全てのデータ値に関し、入力値、即ち「ｏｒｉｇｉｎａｌ」（「元」）値に対応する出力値、即ち「ｒｅｓｕｌｔ」（「結果」）がソフトウェアループにおいて計算される。

result = original + prediction
if result > highValue then result = result - wrapValue

【０１９１】

最終的に、次の入力に関する予測値が、現結果と等しい値に設定され、即ち、以下のとおりとされる。

prediction = result

【０１９２】

エンコーダ１０１０において、第２直接差分演算、即ち方法２が、有利には以下のとおり実施される。全てのデータ値に関し、入力値、即ち「ｏｒｉｇｉｎａｌ」（「元」）値に対応する出力値、即ち「ｒｅｓｕｌｔ」（「結果」）がソフトウェアループにおいて計算される。

result = original - prediction
if result < lowValue then result = result + wrapValue

【０１９３】

最終的に、次の入力に関する予測値が、現結果と等しい値に設定され、即ち、以下のとおりとされる。

prediction = result

【０１９４】

デコーダ１０２０において、第２逆差分演算、即ち方法２が、有利には以下のとおり実施される。全てのデータ値に関し、入力値、即ち「ｏｒｉｇｉｎａｌ」（「元」）値に対応する出力値、即ち「ｒｅｓｕｌｔ」（「結果」）がソフトウェアループにおいて計算される。

result = original + prediction
if result > highValue then result = result - wrapValue

【０１９５】

最終的に、次の入力に関する予測値が、現入力と等しい値に設定され、即ち、以下のとおりとされる。

prediction = original

【０１９６】

エンコーダ１０１０において、第１直接合算演算、即ち方法３が、有利には以下のとおり実施される。全てのデータ値に関し、入力値、即ち「ｏｒｉｇｉｎａｌ」（「元」）値に対応する入力値、即ち「ｒｅｓｕｌｔ」（「結果」）がソフトウェアループにおいて計算される。

result = original + prediction
if result > highValue then result = result - wrapValue

【０１９７】

最終的に、次の入力に関する予測値が、現入力と等しい値に設定され、即ち、以下のと

おりとされる。

prediction = original

【０１９８】

デコーダ１０２０において、第１逆合算演算、即ち方法３が、有利には以下のとおり実施される。全てのデータ値に関し、入力値、即ち「ｏｒｉｇｉｎａｌ」（「元」）値に対応する入力値、即ち「ｒｅｓｕｌｔ」（「結果」）がソフトウェアループにおいて以下のとおり計算される。

result = original - prediction
if result < lowValue then result = result + wrapValue

【０１９９】

最終的に、次の入力に関する予測値が、現結果と等しい値に設定され、即ち、以下のとおりとされる。

prediction = original

【０２００】

エンコーダ１０１０において、第２直接合算演算、即ち方法４が、有利には以下のとおり実施される。全てのデータ値に関し、入力値、即ち「ｏｒｉｇｉｎａｌ」（「元」）値に対応する入力値、即ち「ｒｅｓｕｌｔ」（「結果」）がソフトウェアループにおいて以下のとおり計算される。

result = original + prediction
if result > highValue then result = result - wrapValue

【０２０１】

最終的に、次の入力に関する予測値が、現結果と等しい値に設定され、即ち、以下のとおりとされる。

prediction = original

【０２０２】

デコーダ１０２０において、第２逆合算演算、即ち方法４が、有利には以下のとおり実施される。全てのデータ値に関し、入力値、即ち「ｏｒｉｇｉｎａｌ」（「元」）値に対応する出力値、即ち「ｒｅｓｕｌｔ」（「結果」）がソフトウェアループにおいて以下のとおり計算される。

result = original - prediction
if result < lowValue then result = result + wrapValue

【０２０３】

最終的に、次の入力に関する予測値が、現結果と等しい値に設定され、即ち、以下のとおりとされる。

prediction = original

【０２０４】

このような合算及び差分の演算、４つの方法すべてが、即ちＯＤｅｌｔａバージョンのエンコーダ１０１０及びデコーダ１０２０を実施する際に、１ビットデータ、即ちビット単位で適用可能でもある。１ビットデータの状況においては、次の値は、エンコーダ１０１０及びデコーダ１０２０の両方によって既知であり、つまり、ＭＩＮ＝０、ＭＡＸ＝１である。さらに、有利には、ｌｏｗＶａｌｕｅ＝ＭＩＮ＝０かつｈｉｇｈＶａｌｕｅ＝ＭＡＸ＝１と仮定される。さらに、このような場合、「ｎｕｍｂｅｒ」（「数」）は、従って、［ｈｉｇｈＶａｌｕｅ－ｌｏｗＶａｌｕｅ＝１－０＝１］であり、ｗｒａｐＶａｌｕ

ｅについては、「ｎｕｍｂｅｒ」（「数」）＋１＝１＋１＝２が選択されるという利点を持つ。その理由は、ｌｏｗＶａｌｕｅ＝ＭＩＮ＝０から開始する正の値のみを有し得ると考慮される１ビットデータのみ存在するからである。１ビットデータに関し、方法１及び方法３は、相互に類似の符号化結果を算出する。同様に、方法２及び方法４は、相互に類似の符号化結果を算出する。このような知識を有することは、様々なデフォルトが仮定され得るので、データＤＡ２に送信される必要のある情報を有利に簡素化する。即ち、差分演算、つまり方法１又は方法２の実行回数の数、及び選択された予測（入力値（方法１）又は結果値（方法２））についての情報を送信する必要があるだけで、その結果、デコーダ１０２０は、データＤＡ２、ＤＡ３又はＤＡ４を復号して復号データＤＡ５を生成する際に、正しい逆差分演算を必要回数、実行し得る。

【０２０５】

同様の出力を作成する方法１又は方法３を用いることによって作成された第１の例は、やはり同様の出力を作成する方法２又は方法４を用いることによっても処理され得る。以下に示す結果は、それら方法をデータ式１に適用した場合に取得され得るものである。

０１１００１０００１１１１００１１１１１１１１１１１１１０１０１０１０１０１

【０２０６】

この時、処理データは、２４個の「１」と１３個の「０」とを有し、即ちそのエントロピーは第１の例のものと同じであるが、「１」及び「０」のカウントは位置を変える。これは常に生じる訳ではなく、その代わり、しばしば、これら異なる方法の間でエントロピーが同様に変化する。例えば、データの最初の４つの要素の後、方法１及び／又は方法３は、３個の「１」と１個の「０」を生成するのに対し、元データと、方法２及び／又は方法４を用いて処理されたデータは、２個の「１」と２個の「０」とを有する。従って、このような場合の方法１及び／又は方法３は、方法２及び／又は方法４より小さいエントロピーを生成し、また、元のものよりも小さいエントロピーを生成する。

【０２０７】

マルチビットによる実施においては、データＤＡ１が－６４～＋６３の範囲の値を含む場合、ＭＩＮ＝－６４かつＭＡＸ＝６３である。ｌｏｗＶａｌｕｅ＝ＭＩＮかつｈｉｇｈＶａｌｕｅ＝ＭＡＸと仮定することによって、「ｎｕｍｂｅｒ」（「数」）＝１２７であり、ｗｒａｐＶａｌｕｅには、有利に１２８が選択される。しかしながら、データＤＡ１がランダムに変化する場合、「ｐｒｅｄｉｃｔｉｏｎ」（「予測」）値は、有利には、値［（ｗｒａｐＶａｌｕｅ＋１）÷２＋ｌｏｗＶａｌｕｅ＝６４＋－６４＝０］に設定される。

【０２０８】

第１値が例えば－１である場合、直接ＯＤｃｌｔａ方法１及び／又は方法２を用いた第１符号化値は－１－０＝－１となり、これに対応して、直接ＯＤｅｌｔａ方法３及び／又は方法４を用いた第１符号化値は－１＋０＝－１となるであろう。次の値は、次いで、どのようにデータが進行するかに応じて変化し、例えば第２値が５となる場合、直接ＯＤｅｌｔａ方法１は５－（－１）＝６を生成し、直接ＯＤｅｌｔａ方法２は５－（－１）＝６を生成し、直接ＯＤｅｌｔａ方法３は５＋（－１）＝４を生成し、直接ＯＤｅｌｔａ方法４は５＋（－１）＝４を生成するだろう。デコーダ１０２０は、この場合、逆ＯＤｅｌｔａ方法１及び／又は方法２を用いた場合に第１値として－１＋０＝－１を生成することが可能であり、逆ＯＤｅｌｔａ方法３及び／又は方法４を用いて、－１－０＝－１を生成することが可能であるだろう。これに対応して、逆ＯＤｅｌｔａ方法１を用いた第２値は６＋（－１）＝５となり、逆ＯＤｅｌｔａ方法２を用いた第２値は６＋（－１）＝５となり、逆ＯＤｅｌｔａ方法３を用いた第２値は４－（－１）＝５となり逆ＯＤｅｌｔａ方法４を用いた第２値は４－（－１）＝５となるであろう。

【０２０９】

この解決手段は、数の範囲が実際に－２０～＋２７の値のみを含む場合には、次いで最適化され得る。本例の場合、例えば、ｌｏｗＶａｌｕｅ＝－２０及びｈｉｇｈＶａｌｕｅ

＝２７を伝達することが可能である。両方が伝達される場合は、ｎｕｍｂｅｒ＝４７と計算可能であり、ｗｒａｐＶａｌｕｅには、次いで、有利には４８が選択される。すると、ｐｒｅｄｉｃｔｉｏｎについて、値４８÷２＋ －２０＝４を計算可能である。次いで、以前の例により、ＯＤｅｌｔａ方法１又は方法２が用いられる場合、例えば、値－１に関して－１－４＝－５が生成され、ＯＤｅｌｔａ方法３又は方法４が用いられる場合には、－１＋４＝３が生成される。同様に、第２値は、ＯＤｅｌｔａ方法については、（５－（－１））＝６、（５－（－５））＝１０、（５＋（－１））＝４、及び（５＋３）＝８となるであろう。デコーダ１０２０は、再び正常に機能し、方法１及び／又は方法２に関して第１値を－５＋４＝－１と生成し、方法３及び／又は方法４に関して３－４＝－１を生成する。対応して、各種方法に関する第２値は、（６＋（－１））＝５、（１０＋（－５））＝５、（４－（－１））＝５、及び（８－３）＝５として復号されるだろう。

【０２１０】

上記のこれら例における値は、上記範囲、即ち－６４～＋６３又は－２０～＋２７の内にあることが解り、従って、これらの例による値の内で補正項を実行する必要はないが、任意の負又は正の変化が十分に大きいものである場合には、データ値に対する補正が、与えられる式２１～２４によって為され、上記範囲内に結果値を維持する必要がある。ここで補正項はラップアラウンド値を指していることに留意されたい。

【０２１１】

ｌｏｗＶａｌｕｅが既知である場合、エントロピー符号化データＤＡ３と共にエンコーダ１０１０からデコーダ１０２０に送信されなければならない符号化テーブルを簡素化するために、符号化値は、有利には、０で開始し、かつ値「ｎｕｍｂｅｒ」（「数」）で終了するように構成される。この演算は、ポストオフセットと呼ばれ、このポストオフセット値は、エントロピー符号化の後、かつデータＤＡ４に対する逆ＯＤｅｌｔａ演算の前に、符号化データ値から削除されなければならない。

【０２１２】

当初述べたとおり、ｐｒｅ－ｏｆｆｓｅｔ（プリオフセット）機能を用いてオフセットを実施することも可能であり、この場合、元入力データ（ＤＡ１）は、ＯＤｅｌｔａ方法の実際の実行の前に既に０から「ｎｕｍｂｅｒ」（「数」）までの値を含み得る正の要素に変換される。また、この状況においては、「プリオフセット」及びＯＤｅｌｔａ法が繰り返し同じ情報を伝達しないように、又は何らかの他の方法の結果、既知であることを無視するように、この演算が要する情報伝達が実行されるのが有利である。適切なＤＡ５出力データを作成するためには、このプリオフセットによる効果は、逆ＯＤｅｌｔａ演算の後に復号データから削除されなければならない。

表４： 既知の技術

先行文献	詳細
P1	可変長符号（“Variable-length code”）、ウィキペディア（2012年11月28日にアクセス） URL: http://en.wikipedia.org/wiki/Variable-length_code
P2	ランレングス符号化（“Run-length encoding”）、ウィキペディア（2012年11月28日にアクセス） URL: http://en.wikipedia.org/wiki/Run-length_encoding
P3	ハフマン符号化（“Huffman coding”）、ウィキペディア（2012年11月28日にアクセス） URL: http://en.wikipedia.org/wiki/Huffman_coding
P4	算術符号化（“Arithmetic coding”）、ウィキペディア（2012年11月28日にアクセス） URL: http://en.wikipedia.org/wiki/Arithmetic_coding
P5	通信の数学的理論（“A Mathematic Theory of Communication”）、シャノン・クロード・E（Shannon, Claude E.）（１９４８）、ウィキペディア（2012年11月28日にアクセス） URL: http://cm:bell-labs.com/cm/ms/what/shannonday/shannon1948.pdf
P6	差分符号化（“Delta encoding”）、ウィキペディア（2012年11月28日にアクセス） URL: http://en.wikipedia.org/wiki/Delta_coding
P7	シャノンの情報源符号化定理（Shannon’s source coding theorem）；ウィキペディア（2012年11月28日にアクセス） URL: http://en.wikipedia.org/wiki/Source_coding_theorem
P8	エントロピー（“Entropy”）- ウィキペディア（2012年11月28日にアクセス） URL: http://en.wikipedia/wiki/Entropy

〔付属書類２：ブロックエンコーダの概要〕

【０２１３】

ブロックエンコーダは、英国特許出願公開２５０３２９５号公報に開示されている。この公報の内容は本願明細書の一部を成すものと考えられたい。

【０２１４】

入力データを符号化して対応する符号化出力データを生成する方法が開示されているが、この方法は、

（ａ）前記入力データを複数のブロックまたはパケットに分割することであって、前記ブロックまたはパケットはそれらのコンテンツに応じた、一つ以上のサイズを有する、分割することと；

（ｂ）対応する変換データを生成するために、前記ブロックまたはパケットのコンテンツに対して複数の変換を適用することと；

（ｃ）前記変換データの表現の品質が一つ以上の品質基準を満たすかどうかを決定するために、前記変換データの表現の品質を、変換を適用する前のブロックまたはパケットのコンテンツと比較して調べることと；
（ｄ）一つ以上のブロックまたはパケットに対応する変換データの表現の品質が前記一つ以上の品質基準を満たさない場合、前記一つ以上のブロックまたはパケットをさらに分割および／または結合し、ステップ（ｂ）を繰り返すことと；
（ｅ）一つ以上のブロックまたはパケットに対応する変換データの表現の品質が前記一つ以上の品質基準を満たす場合、符号化される前記入力データを表わす符号化出力データを提供するために、変換データを出力することと；
を含むことを特徴とする。

【０２１５】
また、入力データを符号化して対応する符号化出力データを生成するエンコーダが開示されているが、このエンコーダは、以下の処理：
（ａ）前記入力データを複数のブロックまたはパケットに分割することであって、前記ブロックまたはパケットはそれらのコンテンツに応じた、一つ以上のサイズを有する、分割することと；
（ｂ）対応する変換データを生成するために、前記ブロックまたはパケットのコンテンツに対して複数の変換を適用することと；
（ｃ）前記変換データの表現の品質が一つ以上の品質基準を満たすかどうかを決定するために、前記変換データの表現の品質を、変換を適用する前のブロックまたはパケットのコンテンツと比較して調べることと；
（ｄ）一つ以上のブロックまたはパケットに対応する変換データの表現の品質が前記一つ以上の品質基準を満たさない場合、前記一つ以上のブロックまたはパケットをさらに分割および／または結合し、ステップ（ｂ）を繰り返すことと；
（ｅ）一つ以上のブロックまたはパケットに対応する変換データの表現の品質が前記一つ以上の品質基準を満たす場合、符号化される前記入力データを表わす符号化出力データを提供するために、変換データを出力することと；
を実行するデータ処理ハードウェアを備える。

〔付属書類３：ブロックデコーダの概要〕

【０２１６】
ブロックデコーダは、英国特許出願公開２５０５１６９公報に開示されている。この公報の内容は本願明細書の一部を成すものと考えられたい。

【０２１７】
符号化入力データを復号して対応する復号出力データを生成する方法が開示されているが、この方法は、以下のステップ：
（ａ）前記符号化入力データを処理することであって、前記符号化入力データからヘッダ情報を抽出し、前記ヘッダ情報は、前記符号化入力データに含まれるブロックおよび／またはパケットに関する符号化データを示し、かつ、前記ブロックおよび／またはパケットに関する符号化データとして含めるために、元のブロックおよび／またはパケットを圧縮符号化するのに使用された一つ以上の変換を示す、前記処理することと；
（ｂ）復号ブロックおよび／またはパケットコンテンツを受信するために、データ記憶構成のデータフィールドを用意することと；
（ｃ）前記一つ以上の変換を記述する情報を読み出し、圧縮符号化された元のブロックおよび／またはパケットデータを復号するための、前記一つ以上の変換の逆変換を適用することであって、前記データフィールドに入力するための、対応する復号ブロックおよび／またはパケットコンテンツを生成する、前記読み出し・適用することと；
（ｄ）前記符号化入力データの分割・結合情報に従って、前記データフィールドのブロックおよび／またはパケットを分割および／または結合することと；
（ｅ）前記符号化入力データが少なくとも部分的に復号されると、前記データフィールドから復号出力データとしてデータを出力することと；

を含む。

【０２１８】

また、入力データを復号して対応する復号出力データを生成するように動作するデコーダが開示されているが、このデコーダは、以下の処理：

（ａ）前記符号化入力データを処理することであって、前記符号化入力データからヘッダ情報を抽出し、前記ヘッダ情報は、前記符号化入力データに含まれるブロックおよび／またはパケットに関する符号化データを示し、かつ、前記ブロックおよび／またはパケットに関する符号化データとして含めるために、元のブロックおよび／またはパケットを圧縮符号化するのに使用された一つ以上の変換を示す、前記処理することと；

（ｂ）復号ブロックおよび／またはパケットコンテンツを受信するために、データ記憶構成のデータフィールドを用意することと；

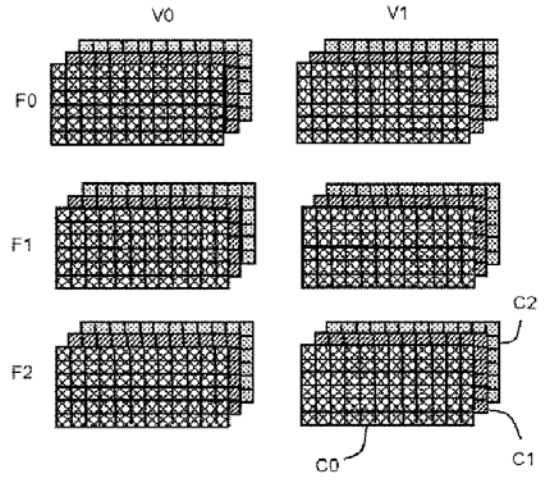
（ｃ）一つ以上の変換を記述する情報を読み出し、圧縮符号化された元のブロックおよび／またはパケットデータを復号するための、一つ以上の変換の逆変換を適用することであって、データフィールドに入力するための、対応する復号ブロックおよび／またはパケットコンテンツを生成する、前記読み出し・適用することと；

（ｄ）符号化入力データの分割・結合情報に従って、データフィールドのブロックおよび／またはパケットを分割および／または結合することと；

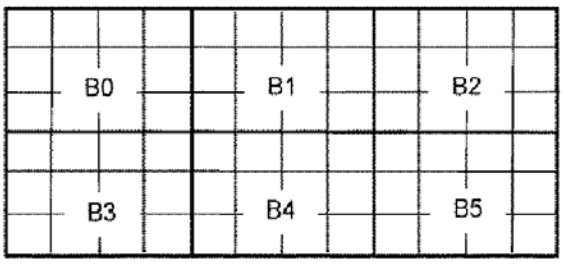
（ｅ）符号化入力データが少なくとも部分的に復号されると、データフィールドから復号出力データとしてデータを出力することと；

を実行するデータ処理ハードウェアを備える。

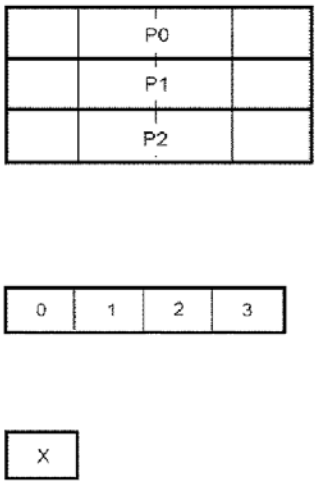
【図１】



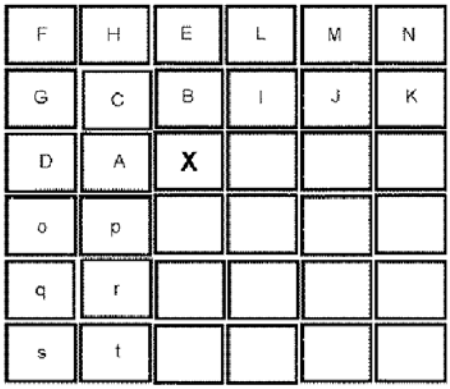
【図２】



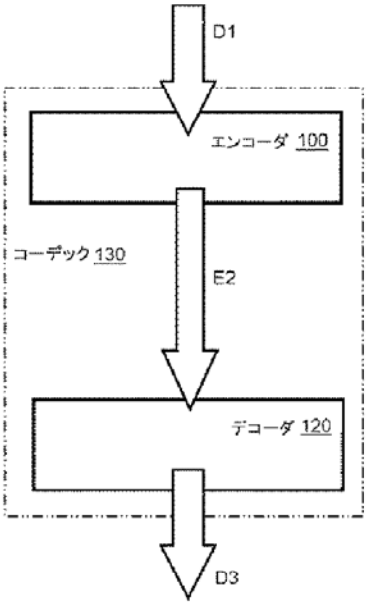
【図３】



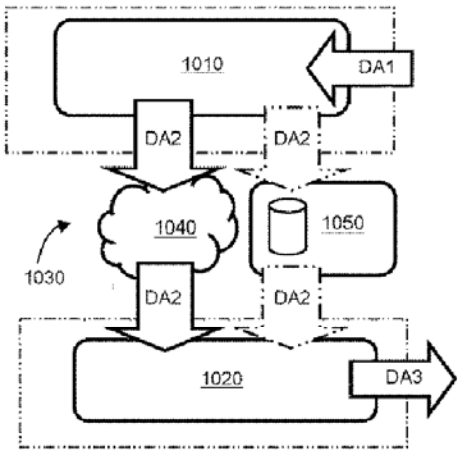
【図４】


F
H
E
L
M
N
G
C
B
I
J
K
D
A
X
o
p
q
r
s
t

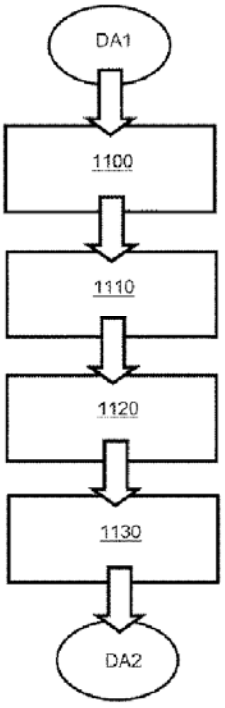
【図５】


D1
エンコーダ 100
コーデック 130
E2
デコーダ 120
D3

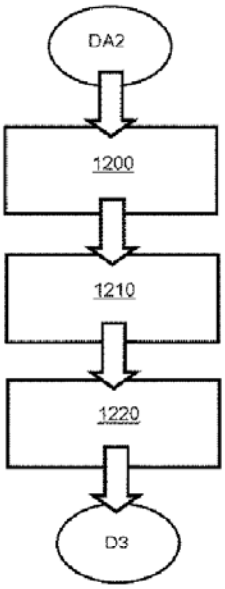
【図６】


1010
DA1
DA2
DA2
1040
1050
1030
DA2
DA2
1020
DA3

【図７】


DA1
1100
1110
1120
1130
DA2

【図８】



フロントページの続き

(51)Int.Cl. F I
Ｈ０４Ｎ 5/926 (2006.01) Ｈ０４Ｎ 5/926 ２００

審査官 久保 光宏

(56)参考文献 特許第４２２３５７７（ＪＰ，Ｂ２）
国際公開第２０１１／０２４６０２（WO，Ａ１）
国際公開第２０１３／１７８５２４（WO，Ａ１）
米国特許第５８１５２０７（ＵＳ，Ａ）
特許第６０４５１２３（ＪＰ，Ｂ２）
大久保榮監修，「インプレス標準教科書シリーズ 改訂三版H.264/AVC教科書」，日本，株式会社インプレスＲ＆Ｄ，２００９年 １月 １日，第１版，第18,19,28,29,86-89,97,98,110-116頁，ISBN: 978-4-8443-2664-9
杉山賢二，「実践 映像信号処理 －Ｃ言語を使って体感する－」，日本，株式会社コロナ社，２００８年 ３月１８日，初版，第１４～１６，３９，４０頁，ISBN: 978-4-339-00794-7
David Salomon, "Data Compression", Springer, ２００７年, Fourth Edition, Pages 47-56,112-119,444-454,676-718, ISBN: 1-84628-602-6, [online], [令和元年6月24日検索]，インターネット，ＵＲＬ，http://read.pudn.com/downloads167/ebook/769449/dataCompress.pdf

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｈ０４Ｎ１９／００－１９／９８
Ｈ０４Ｎ５／９２６
ＣＳＤＢ（日本国特許庁）
ＩＥＥＥＸｐｌｏｒｅ（ＩＥＥＥ）