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

特許第5036526号
(P5036526)

(45)発行日　平成24年9月26日(2012.9.26)　(24)登録日　平成24年7月13日(2012.7.13)

(51)Int.Cl.			F I		
H04N	1/46	(2006.01)	H04N	1/46	Z
H04N	1/60	(2006.01)	H04N	1/40	D
G06T	1/00	(2006.01)	G06T	1/00	510

請求項の数 5　(全 33 頁)

(21)出願番号　特願2007-337660 (P2007-337660)
(22)出願日　平成19年12月27日 (2007.12.27)
(65)公開番号　特開2009-159493 (P2009-159493A)
(43)公開日　平成21年7月16日 (2009.7.16)
審査請求日　平成22年12月20日 (2010.12.20)

(73)特許権者　000001007
キヤノン株式会社
東京都大田区下丸子３丁目３０番２号
(74)代理人　100076428
弁理士　大塚　康徳
(74)代理人　100112508
弁理士　高柳　司郎
(74)代理人　100115071
弁理士　大塚　康弘
(74)代理人　100116894
弁理士　木村　秀二
(74)代理人　100130409
弁理士　下山　治
(74)代理人　100134175
弁理士　永川　行光

最終頁に続く

(54)【発明の名称】変換テーブル圧縮方法およびプログラム

(57)【特許請求の範囲】

【請求項１】

色変換テーブルを圧縮して記憶媒体に格納するための色変換テーブル圧縮方法であって、

互いに同じ構成を持つ少なくとも２つの色変換テーブルの同一の格子点に対応する値を、前記格子点ごとに一定の順序で配置するマージ処理を、前記色変換テーブルの各格子点に対して行ったマージテーブルを得るマージステップと、

前記マージテーブルに格納されているマージ処理されたデータを圧縮して圧縮マージテーブルとする圧縮ステップと、

前記マージテーブルが前記少なくとも２つの色変換テーブルのデータをマージ処理して得られていることを示す情報と前記圧縮マージテーブルとを記録媒体に格納する格納ステップと

を有したことを特徴とする色変換テーブル圧縮方法。

【請求項２】

色変換テーブルを圧縮して記憶媒体に格納するための色変換テーブル圧縮方法であって、

互いに同じ構成を持つ少なくとも２つの色変換テーブルに格納されているデータを、当該色変換テーブルの入力空間を規定する複数の軸それぞれの方向について、隣接する格子間の差分値にそれぞれ変換して少なくとも２つの全軸差分テーブルとする全軸変換ステップと、

前記少なくとも２つの全軸差分テーブルの同一の格子点に対応する値を、前記格子点ごとに一定の順序で配置するマージ処理を、前記全軸差分テーブルの各格子点に対して行ったマージ全軸差分テーブルを得るマージステップと、
当該マージ全軸差分テーブルに格納されているマージ処理されたデータを圧縮して圧縮マージ全軸差分テーブルとするステップと、
当該マージ全軸差分テーブルが前記少なくとも２つの全軸差分テーブルのデータをマージ処理して得られていることを示す情報と当該圧縮マージ全軸差分テーブルを記録媒体に格納するステップと
を有したことを特徴とする色変換テーブル圧縮方法。
【請求項３】
色変換テーブルを圧縮して記憶媒体に格納するための色変換テーブル圧縮方法であって、
互いに同じ構成を持つ少なくとも２つの第１の色変換テーブルを構成する要素を間引いて、データ量の少ない第２の色変換テーブルをそれぞれ作成する間引きステップと、
前記少なくとも２つの第１の色変換テーブルの要素のうち前記第２の色変換テーブルに含まれない要素の値を、前記少なくとも２つの第１の色変換テーブルの要素と前記第２の色変換テーブルを補間して得た要素値との差分値にそれぞれ変換する差分ステップと、
前記少なくとも２つの第２の色変換テーブルと前記差分値とをそれぞれ連結して連結テーブルとする連結ステップと、
前記少なくとも２つの連結テーブルの同一の格子点に対応する値を、前記格子点ごとに一定の順序で配置するマージ処理を、前記連結テーブルの各格子点に対して行ったマージ連結テーブルを得るマージステップと、
当該マージ連結テーブルに格納されているマージ処理されたデータを圧縮して圧縮マージ連結テーブルとするステップと、
当該マージ連結テーブルが前記少なくとも２つの連結テーブルのデータを互いにマージ処理して得られていることを示す情報と、当該圧縮マージ連結テーブルとを記録媒体に格納するステップと
を有したことを特徴とする色変換テーブル圧縮方法。
【請求項４】
請求項１乃至３のいずれか１項に記載された色変換テーブル圧縮方法を実行して色変換テーブルを圧縮することを特徴とする色変換テーブル圧縮装置。
【請求項５】
コンピュータにより、請求項１乃至３のいずれか１項に記載された色変換テーブル圧縮方法を実行するためのプログラム。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、たとえば画像の色変換等に用いられる変換テーブルの圧縮方法に関する。詳しくは、カラー画像処理装置で画像データの処理に用いる変換テーブル、特に色変換テーブルを格納するために必要な記憶容量を小さくするための方法、および、変換テーブルを格納に適したデータに変換する。または、格納した変換テーブルを用いて画像処理を行なう画像処理装置に関する。
【背景技術】
【０００２】
カラープリンタやカラー複写機またはカラースキャナなどのカラー画像の入力あるいは出力を行う機器では、高品位なカラー画像を実現するために、人間の知覚特性などに基づいて高度に最適化された色変換処理を行う必要がある。このような色変換処理は、単純な演算によって実現することが困難であるため、自由度の高い変換処理を実現することが可能であるルックアップテーブル（以下、ＬＵＴとする）を用いた変換処理により実施されている。ＬＵＴとは変換テーブルの一種であり、入力に対応する出力値をテーブルとして

保持する。ＬＵＴに考えられる全ての入力に対する出力データを保持するためには膨大な記憶容量が必要となる。そのため、ＬＵＴには、離散的に定めた特定の入力（格子点という）に対する出力データ（格子点値という）を保持し、格子点以外の入力に対する出力データは、格子点値を使用して補間により算出されることが一般的である。

【０００３】

例えば、色変換処理として最も一般的である、入力データがＲ（レッド）、Ｇ（グリーン）、Ｂ（ブルー）各色８ビットで構成され、出力データがＣ（シアン）、Ｍ（マゼンタ）、Ｙ（イエロー）、Ｋ（ブラック）各色８ビットとなる色変換処理を考える。８ビットは２５６階調に相当する。この変換に用いられるＬＵＴは、ＲＧＢを座標軸とする３次元のテーブルとなり、ＲＧＢの値に応じたＹＭＣＫの値をその要素として含む。この場合入力データ（Ｒ，Ｇ，Ｂ）がとり得る値は、０ｈ～ＦＦＦＦＦＦｈまでの約１６７０万通りとなる。したがってこの色変換のために用いるＬＵＴには、約１６７０万×４バイト＝約６４Ｍバイトのデータを記憶する必要ある。

【０００４】

そこで、データ量を削減するために、特定の入力、例えばＲＧＢの各値が０、６４、１２８、１９１、２５５　のいずれかであるような値を格子点とし、ＬＵＴにはその格子点に対応するＹＭＣＫの値を格子点値として保存する。たとえば、格子点となる入力データは、（０、０、０）、（１２８、１９１、６４）、（６４、２５５、１２８）などであり、それら格子点に対応する格子点値すなわち出力データがＬＵＴに保持される。ここで改めて定義すると、出力データがＬＵＴに保持されている入力データを格子点、保持された出力データの値をその格子点値という。そして格子点以外の入力データ、たとえば（１２８、２００、６４）などに対する出力データは、格子点値を格子点からの距離に応じた加重平均により算出することが一般的には実施されている。このように格子点値を加重平均することにより出力データを算出することは補間と呼ばれ、様々な補間の手法が提案されている（特許文献４等参照）。

【０００５】

なお各軸の格子点の数を格子点数と呼び、前述の例では格子点数は５である。また、各軸の格子点が対応する入力データの値、前述の例では０、６４、１２８、１９１、２５５

を格子位置と呼ぶ。また格子位置をその値が小さいものから順につけた番号を格子番号と呼ぶ。例えば前述の例では格子位置０の格子番号は０、格子位置１９１の格子番号は３となる。なお、前述の例では各軸の格子点数、格子位置がすべて同じとしたが、格子点数も格子位置も各軸ごとに異なるものにすることも可能である。ＬＵＴでは内挿により格子点でない入力データに対応する出力データを算出するため、少なくとも入力データの最小値と最大値に対応する格子点は保持する必要があるので格子点数の最小値は２となる。

【０００６】

近年、インクジェットプリンタをはじめとする画像処理装置は銀塩写真を超えるほどの高画質の実現するようになり、その用途をさらに広げるためより多くの用紙の種類に対応することが求められてきている。また、更なる高画質を実現するためにより高精度な色変換処理の実現が求められている。より多くの用紙の種類に対応するためには、それぞれの用紙に最適化された色変換テーブルが必要であるため１つの画像処理装置において使用される色変換テーブルの数はさらに増えている。また、テーブルに補間演算を併用した色変換処理を高精度に行おうとする場合、その色変換テーブルの格子点の数を多くし、補間演算で用いる格子点と入力データの距離を小さくすることが有効な手法の一つである。しかし、格子点数が多くなると１つのＬＵＴのデータ量が大幅に増加することになる。これらの要因により色変換テーブルのデータ量はますます増大し、そのデータを格納するために使用されるメモリー等のリソースを圧迫する結果をもたらしている。

【０００７】

一方、限られたリソースに相対的に多量のデータを収める技術として、データ圧縮が知られている。しかしながら、既知のデータ圧縮を色変換テーブルの圧縮にそのまま適用しても、十分にデータ量の低減が図れない、つまり高い圧縮率が得られない問題があった。

これは以下の理由による。
【０００８】
すなわち、色の連続性により色変換テーブルは通常滑らかな、しかし非線形に変化する値の集合として表現されるため、色変換テーブルを走査して得られるデータ列の値も滑らかに変化する。しかしその変化は規則的なものではなく、さらにその値は広い範囲に偏ることなく分布しているからである。
【０００９】
一方、エントロピー符号化を用いた通常の圧縮処理では、同じデータが高い頻度で繰り返し出現すること利用してデータ量の削減を図っている。このため、同じデータが高頻度で表れるデータの圧縮率は高く、そうでないデータの圧縮率は低くなる。色変換テーブルは、色の連続性により、通常滑らかな、しかし非線形に変化する値の集合として表現される。そのため、色変換テーブルの値は入力の変化に応じて滑らかに変化するが、その変化は規則的なものではなく、さらにその値は広い範囲に偏ることなく分布している。このために、既知のさまざまな圧縮アルゴリズムのどのようなものを適用しても色変換テーブルの圧縮率を挙げることは困難であるという問題がある。
【００１０】
そこで圧縮処理を施す前に色変換テーブルに対して変換処理を施すことで色変換テーブルの圧縮率を向上する手法が提案されている（たとえば特許文献１乃至特許文献３等参照）。
【特許文献１】特開平１１－０１７９７１号公報
【特許文献２】特開２００２－２０９１１４号公報
【特許文献３】特開２００３－１１０８６９号公報
【特許文献４】特開昭５３－１２３２０１号公報
【発明の開示】
【発明が解決しようとする課題】
【００１１】
しかし、既に提案されている色変換テーブルに対して圧縮処理の前に変換処理を施す手法には、いくつもの欠陥があった。たとえば、（１）圧縮率の向上が十分なものではない。（２）色変換テーブルが一定の条件を満たさないとその手法を適用することができない。（３）個々の色変換テーブルを個別に処理しているため色変換テーブルの数が増加した場合にも個々の色変換テーブルの圧縮率は変化せず、そのため色変換テーブルの数の増加と同じ割合で圧縮後の色変換テーブルの合計データ量も増加してしまう。
【００１２】
上記欠陥を解消して変換テーブルの容量を縮小するために、より効率的で汎用的な圧縮処理の手法が必要とされている。
【００１３】
本発明は、上述の従来例に鑑みてなされたものであり、その目的とするところは、上記欠陥を解消することにある。より具体的には、色変換テーブルをデータ圧縮して記憶媒体に格納する場合に、どのような色変換テーブルにも適応することが可能で、色変換テーブルの数が増加するに従って圧縮率を向上させることが可能である色変換テーブル圧縮方法を提供することにある。
【課題を解決するための手段】
【００１４】
上述の問題を解決するための本発明は以下の構成を備える。すなわち
色変換テーブルを圧縮して記憶媒体に格納するための色変換テーブル圧縮方法であって、
互いに同じ構成を持つ少なくとも２つの色変換テーブルの同一の格子点に対応する値を、前記格子点ごとに一定の順序で配置するマージ処理を、前記色変換テーブルの各格子点に対して行ったマージテーブルを得るマージステップと、
前記マージテーブルに格納されているマージ処理されたデータを圧縮して圧縮マージテ

ーブルとする圧縮ステップと、
前記マージテーブルが前記少なくとも２つの色変換テーブルのデータをマージ処理して得られていることを示す情報と前記圧縮マージテーブルとを記録媒体に格納する格納ステップとを有した。
【００１５】
あるいは、色変換テーブルを圧縮して記憶媒体に格納するための色変換テーブル圧縮方法であって、
互いに同じ構成を持つ少なくとも２つの色変換テーブルに格納されているデータを、当該色変換テーブルの入力空間を規定する複数の軸それぞれの方向について、隣接する格子間の差分値にそれぞれ変換して少なくとも２つの全軸差分テーブルとする全軸変換ステップと、
前記少なくとも２つの全軸差分テーブルの同一の格子点に対応する値を、前記格子点ごとに一定の順序で配置するマージ処理を、前記全軸差分テーブルの各格子点に対して行ったマージ全軸差分テーブルを得るマージステップと、
当該マージ全軸差分テーブルに格納されているマージ処理されたデータを圧縮して圧縮マージ全軸差分テーブルとするステップと、
当該マージ全軸差分テーブルが前記少なくとも２つの全軸差分テーブルのデータをマージ処理して得られていることを示す情報と当該圧縮マージ全軸差分テーブルを記録媒体に格納するステップとを有した。
【００１６】
あるいは、色変換テーブルを圧縮して記憶媒体に格納するための色変換テーブル圧縮方法であって、
互いに同じ構成を持つ少なくとも２つの第１の色変換テーブルを構成する要素を間引いて、データ量の少ない第２の色変換テーブルをそれぞれ作成する間引きステップと、
前記少なくとも２つの第１の色変換テーブルの要素のうち前記第２の色変換テーブルに含まれない要素の値を、前記少なくとも２つの第１の色変換テーブルの要素と前記第２の色変換テーブルを補間して得た要素値との差分値にそれぞれ変換する差分ステップと、
前記少なくとも２つの第２の色変換テーブルと前記差分値とをそれぞれ連結して連結テーブルとする連結ステップと、
前記少なくとも２つの連結テーブルの同一の格子点に対応する値を、前記格子点ごとに一定の順序で配置するマージ処理を、前記連結テーブルの各格子点に対して行ったマージ連結テーブルを得るマージステップと、
当該マージ連結テーブルに格納されているマージ処理されたデータを圧縮して圧縮マージ連結テーブルとするステップと、
当該マージ連結テーブルが前記少なくとも２つの連結テーブルのデータを互いにマージ処理して得られていることを示す情報と、当該圧縮マージ連結テーブルとを記録媒体に格納するステップとを有した。
【発明の効果】
【００１７】
本発明により上述した欠点を解消することができる。詳しくは、本発明により、様々な形態で変換テーブルの格子点値を差分データ化することになり、取りうる範囲全体に分散していた格子点値の大部分が０近辺に偏ることになる。このため、エントロピー符号化を用いた場合に大幅に圧縮率を向上させることができる。特にＬＵＴ内での格子点値の差分化と２つ以上のＬＵＴでのマージを合わせて行うことにより、ＬＵＴの数が増加した場合にもデータ量の増大を効果的に抑えることが可能になる。
【発明を実施するための最良の形態】
【００１８】
以下、図面を参照して本発明の実施形態を詳細に説明する。まず、すべての実施形態において共通するルックアップテーブル（以下ＬＵＴ）の構成について説明する。ＬＵＴは３つの要素から構成される入力データに対して１つ以上のデータを出力する３次元テーブ

ルである。このような３次元テーブルは、図１に示すように直方体として模式的に表現することができる。直方体における１つの角を原点とすると、その角を形成する３つの辺が入力軸となり、入力データの３つの要素の値が角からの距離と対応する。各格子点には格子点値が格納される。このように変換テーブル１は、規則的にサンプリングした入力データに対する出力データである格子点値を格納している格子点１００から構成されている。図１では省略してあるが直方体のかくれている３面および直方体の内部にも規則的に格子点が並んでおり、全体で１０５個の格子点から構成されている。そして以下の実施形態では、変換テーブルは、ＲＧＢ３成分の輝度データを入力とし、ＣＭＹＫ４成分の濃度データを出力とする色変換テーブルである。入力データとしてはＲＧＢのほか、ＣＭＹ、Ｌａｂ、ＸＹＺなどの入力色空間で定義されたデータが想定される。しかし、それらの以外であっても３つの要素のいずれもが有限の範囲をとるデータであれば本発明を適応することができる。また、出力色空間もＹＭＣＫ色空間に限られない。さらに、たとえばＹＭＣＫからＲＧＢへ変換するための色テーブルのように、入力色空間が３次元ではない色空間であり、出力色空間が４次元ではない色空間であっても、各実施形態は適用可能である。さらに、本実施形態は、色変換テーブルと同様に、入力データの変化に応じて出力値が滑らかに変化するが、変化が非線形であり、圧縮しても高圧縮率が得られないような変換テーブル一般に対して適用することもできる。以下では入力データをＲＧＢ、出力データをＹＭＣＫとする色変換テーブルについて説明する。

【００１９】

図１に示した３次元テーブルは、図２のテーブル２０１～２０３に示すように、入力データのうち１つの要素を格子位置ごとに分割し、残りの２つの要素を２次元的に配置して、それぞれを縦軸、横軸に対応づけることでも模式的に表現することができる。この変換テーブルは、出力データの成分をおのおの独立して保持している。つまり３要素からなる入力により１つの値を出力する変換テーブルである。例えばＲＧＢを入力としＣＭＹＫの４つの値を出力する変換テーブルは、ＲＧＢ値に応じてＣＭＹＫのそれぞれの値を与える４つの変換テーブルを１組としたものである。したがって、たとえばＣの値は入力ＲＧＢの値によってのみ決められる、つまりＣ＝Ｆｃ(Ｒ、Ｇ、Ｂ)である（ＦｃはＣ成分の変換テーブルによる写像）。同様にＭ＝Ｆｍ(Ｒ、Ｇ、Ｂ)、Ｙ＝Ｆｙ(Ｒ、Ｇ、Ｂ)、Ｋ＝Ｆｋ(Ｒ、Ｇ、Ｂ）である。このようなＬＵＴは、通常、各色成分のＬＵＴの格子位置は共通である。本実施形態の処理は、処理の対象となる変換テーブルの出力データの次元数がいくつであっても、各成分について個別に適応できる。そのため、以下の説明では出力データを１次元とする。

【００２０】

以下の説明では、特にＣ[ｉ:ｊ:ｋ]という名のＬＵＴを例にして説明する。また、格子点値は全て正の値で８ビットで表されるものとする。各軸の格子番号を決めれば３次元空間での格子位置が確定される。[ｉ:ｊ:ｋ]はＲ、Ｇ、Ｂの格子番号がそれぞれｉ、ｊ、ｋである格子点を表し、Ｃ[ｉ:ｊ:ｋ]は３軸の格子番号がそれぞれｉ、ｊ、ｋである格子点値を表すものとする。また、Ｒ、Ｇ、Ｂの各軸の格子点数をそれぞれＮ、Ｍ、Ｌとする。視覚的な理解を容易にするために、格子点を３次元に配置した模式図を図１に示したが、ＬＵＴを汎用的なデジタル演算処理装置により処理する場合には１次元のテーブルとする必要がある。ＬＵＴを１次元テーブルＣ[ｘ]により表す手法は何通りか存在する。本実施形態では、１次元テーブルにおける位置と、ＬＵＴの格子点を図１のように３次元的に配置したときの３次元空間における格子点の隣接関係が最も容易に関係づけられるように、図３に示すような規則によりＬＵＴを１次元テーブル化する。この規則ではＣ[ｉ:ｊ:ｋ]の格子点値は、１次元テーブルではＣ[ｉ*Ｍ*Ｌ+ｊ*Ｌ+ｋ]となる。図３は一例としてＮ＝５、Ｍ＝３、Ｌ＝７の場合のＣ[ｘ]とＣ[ｉ:ｊ:ｋ]との関係を示している。もちろんＣ[ｘ]とＣ[ｉ:ｊ:ｋ]との相違は本質的なものではなく、ＬＵＴの要素の表現方法の相違にすぎない。したがって本実施形態ではＬＵＴの添え字を一次元の値で説明しているが、３次元で表現した場合でも同様な処理を行える。図３においては、変換テーブルＣ[ｘ]においては、インデックスｘ＝ｉ*Ｍ*Ｌ+ｊ*Ｌ+ｋにより特定された要素がアクセスされる。

【００２１】
１つの格子点が複数の格子点値を保持するＬＵＴの場合は、各格子点値ごとに図４のように第１要素の１次元テーブルの次に第２要素の１次元テーブルを、第２要素の１次元テーブルの次に第３要素の１次元テーブルを圧縮処理の対象とする。このように各要素のテーブルを連結して１つのデータとし、このデータ全体を圧縮処理の対象とする。図４においては、まず入力座標系の各軸の格子点位置４０１が記述され、次にシアン要素の変換テーブル４０２が図３の順序で記述され、同様にマゼンタ、黄、黒の各出力要素４０３～４０５について記述される。格子点位置４０１には、格子点に対応する入力値が好ましくは昇べき又は降べきに、各格子点毎に区別できるように記述されている。たとえば格子点が、０，１２８，２５５であれば、この値が区別可能に記述される。もちろん格子位置は各軸ごとに記述される。なお、図４の例では、全出力要素について格子位置が共通である。もしも共通でなければ、ＲＧＢの格子点位置が、各色成分の変換テーブルごとに用意される。

【００２２】
なお、以下の説明において具体的な例を示す場合には図３に示した格子点数がＲ軸、Ｇ軸、Ｂ軸それぞれ５、３、７のＬＵＴ（総格子点数１０５）を例とする。もちろん本発明は２以上のいかなる格子点数のＬＵＴに対しても適応可能である。

【００２３】
さて、個別の実施形態について説明する前に、本実施形態に示すフローチャートに即したプログラムを実行するコンピュータの構成を説明する。コンピュータは通常の汎用コンピュータである。ホスト装置５１と画像出力装置５２のハードウェア構成について図２９を参照して説明する。図２９は情報処理システムを構成するホスト装置（コンピュータ）５１と画像出力装置（プリンタ）５２のハードウェア構成概要を示すブロック図である。図２９に示されているように、ホスト装置５１は処理部１０００とこれに周辺装置を含めてホスト装置全体が構成されている。また、プリンタ５２は、エンジン部および制御回路部から構成されている。

【００２４】
ホスト装置５１の処理部１０００は、制御プログラムに従ってホスト装置の全体制御を司るＭＰＵ１００１、システム構成要素を互いに接続するバス１００２、ＭＰＵ１００１が実行するプログラムやデータ等を一時記憶するＤＲＡＭ１００３を含む。さらに、システムバスとメモリバス、ＭＰＵ１００１を接続するブリッジ１００４、例えば、ＣＲＴなどの表示装置２００１にグラフィック情報を表示するための制御機能を備えたグラフィックアダプタ１００５を含んでいる。さらに、処理部１０００はＨＤＤ装置２００２ａおよびリムーバブルディスク２００２ｂとのインタフェースを司るディスクコントローラ１００６を有する。さらに、キーボード２００３とのインタフェースを司るキーボードコントローラ１００７、ＩＥＥＥ１２８４あるいはＵＳＢ規格に従って画像出力装置５２との間の通信を司る、インタフェースである通信Ｉ／Ｆ１００８を備えている。さらに、処理部１０００には、グラフィックアダプタ１００５を介して操作者にグラフィック情報等を表示する表示装置２００１（この例では、ＣＲＴ）が接続されている。更に、プログラムやデータが格納された大容量記憶装置であるハードディスクドライブ（ＨＤＤ）装置２００２ａ、キーボード２００３が夫々、コントローラを介して接続されている。

【００２５】
各実施形態で説明する圧縮された色変換テーブルは、たとえば不図示のネットワークを介してサーバからダウンロードされたり、あるいはリムーバブルディスクの媒体２００２ｂにより電磁データとして提供される。そして提供された色変換テーブルは、ＨＤＤ２００２ａに保存される。コンピュータ５１では、プリンタ５２に対して送信する画像データの変換等を行うプリンタドライバが実行される。ＨＤＤ２００２ａに格納された色変換テーブルはこのプリンタドライバにより参照されて、ＲＧＢ色空間で定義された色から、ＹＭＣＫ色空間で定義された色への色変換が実行される。ここでＨＤＤ２００２ａには、種々の印刷設定に対応して複数の色変換テーブルが圧縮された状態で保存されている。また

プリンタドライバは、用紙の種類等の印刷設定を利用者が指定することができるユーザインターフェースを提供している。プリンタドライバは、指定された印刷設定に対応する色変換テーブルを、圧縮された複数の色変換テーブルから選択して復号（本例では、圧縮に対する用語として復号を用いる。伸長と呼ぶ場合もある）する。そして復号した色変換テーブルを色変換のために使用する。

【００２６】

[第１実施形態]

＜色変換テーブルの圧縮処理＞

図５Ｂは、図１の色変換テーブルを圧縮するフローチャートである。図５Ｂの手順は本例では、たとえば色変換テーブルを提供するベンダのコンピュータにより実行される。まず、色変換テーブルの第１の出力色成分に着目して（Ｓ１０３）、その着目色成分のＬＵＴ変換処理を行う（Ｓ１０４）。本例では出力色がＹＭＣＫであるから、たとえばＣに着目し、Ｃ色成分について、色変換テーブルの変換を行う。そして全色成分に変換が終了したかテストし（Ｓ１０５）、終了していなければ次の色成分に着目して（Ｓ１０６）、ＬＵＴ変換を行う。一方全色成分にＬＵＴ変換が終了していれば、色変換テーブル全体について、可逆圧縮処理を実行し（Ｓ１０７）、圧縮データを記録媒体に保存する（Ｓ１０８）。圧縮した色変換テーブルをたとえばネットワークを介してオンラインで頒布する場合には、記録媒体はＨＤＤ２００２ａである。また、蓄積媒体を介してオフラインで頒布する場合には、記録媒体はリムーバブル媒体である。もちろんＨＤＤ２００２ａからリムーバブル媒体へと複製しても良い。

【００２７】

＜色変換テーブルの前置変換処理＞

図５Ａは、本発明の第１実施形態にかかるＬＵＴの変換処理の手順を示すメインフローチャートである。図５ＢのステップＳ１０４において圧縮に先立って実行される。本実施形態のＬＵＴに対する処理は大きく分けてＳ１００、Ｓ１０１、Ｓ１０２の３つのステップに分けられる。各ステップでは、ＲＧＢの各軸方向についての軸方向の差分を求める処理をそれぞれ行う。ステップＳ１００の詳細な処理手順を図６のフローチャートに示す。図６による処理対象は、図４の色変換テーブルである。各色成分のテーブルは、図３のように１次元のインデックスでアクセスされる。

【００２８】

図５ＡのステップＳ１００では、ある格子点Ｘの格子点値と、第３軸方向について格子番号が格子点Ｘよりも一つ小さい隣接格子点の格子点値との差分を求め、その差分を、格子点Ｘの格子点値として代入する処理を、すべての格子点について行う。なお、３次元の入力色空間の各軸を第１軸、第２軸、第３軸と呼び、本例では、ＲＧＢ色空間のＢ成分が第３軸に相当する。

【００２９】

図６において、ステップＳ１１０からＳ１１６で、第３軸の格子番号０番の格子点値を除く全ての格子点値を、１つ前の格子点値との差分値とする。ここで注意が必要なのは、差分値とした値を元の格子点値が格納されていた領域に上書きしている点である。このように上書きをせずに差分値を別の領域に格納して処理を行うことも可能であるが、そのような処理では処理対象のＬＵＴの格納領域とは別に差分値を格納するための同じ量の格納領域が必要になる。逆方向に処理を進めることにより差分値を上書きしながらの処理が可能になる。

【００３０】

図６においては、Ｎ，Ｍ，Ｌは色変換テーブルの格子数に応じて決まる定数であり、図４のＲＧＢ各色成分の格子点の数がそれぞれＮ，Ｍ，Ｌに代入される。また、Ｃ［ｘ］は、図３のように一次元のインデックスで示した形式の色変換テーブルであり、図６（図７，図８も同様）ではシアン成分を例とした。ｐ，ｑは変数であり、ＤＲＡＭ１００３に変数として確保される。

【００３１】

変数ｐは第２軸（すなわちＧ）方向についての格子点位置を示す。１次元で格子点位置（インデックス）ｘを与える式ｘ＝ｉ*M*Ｌ+ｊ*Ｌ+ｋ＝（ｉ*M+ｊ）*Ｌ+ｋに即して説明すると、ｐ＝（ｉ*M+ｊ）*Ｌに相当する。ＭおよびＬは定数であり、ｐはＬを単位として増加するから、ｐが与えられれば第１軸方向の格子位置ｉと第２軸方向の格子位置ｊは定まる。また変数ｑはｋに相当する。すなわちｑにより第３軸方向の格子位置が定まる。ステップＳ１１３～Ｓ１１５では、変数ｐで定まる第１軸および第２軸の格子位置において、第３軸すなわちｑの値をまず最大値すなわちＬ－１とする。その格子点を着目格子点として、着目格子点の格子点値から、当該第３軸上で着目格子点に隣接する格子点の格子点値を差し引き、その値を着目格子点の値として上書きする。ただしここで隣接する格子点とは、着目格子点よりも格子位置がちいさい方向に隣接する格子点である。これを、着目格子位置を、第３軸上で１ずつ小さくしながら繰り返す。ｋ＝ｑ＝０の格子位置は差分をとる相手が存在しないので、ｑ＞０についてのみ行われる。そしてステップＳ１１６では、差分計算する格子位置を、第２軸方向に１ずらす。

【００３２】

この処理は、ＬＵＴを図１のような格子点を三次元的に配置した直方体とした場合にＢ軸方向に隣り合う格子点値の差分を算出することに対応する。この差分処理により格子点値がＢ軸方向の変化量になる。

【００３３】

ここで本実施形態での差分を算出する処理について説明する。通常、２つの値ａ、ｂの差分ｄはｄ＝ａ－ｂとして、ａ＞＝ｂなら正、ａ＜ｂなら負という符号付きの値として算出される。そのように演算した場合、ａおよびｂが８ビットの０～２５５の値であっても差分ｄは－２５５～２５５となり、符号付きの正しい差を記録するために符号ビットを含めた９ビットが必要となる。しかし、本発明における差分では２つの値の正しい差は必要ではなく、復元が可能であれば十分である。そこで差分処理を次にプログラミング言語であるＣ言語で示す処理のように行うことにより、差分ｄを０～２５５、つまり８ビットで表せる範囲の値とすることができる。

```
ｄ＝ａ－ｂ；
ｉｆ（ｄ＜０）{
  ｄ＝ｄ＋２５６；
}。
```

【００３４】

また差分ｄとｂの値からａの値を復元する併合処理は次のように行う。

```
ａ＝ｂ＋ｄ；
ｉｆ（ａ＞＝２５６）{
  ａ＝ａ－２５６；
}。
```

【００３５】

これら演算は、８ビットを越える桁数で演算した場合処理であって、たとえば演算の桁数を８ビットに限れば、符号（サイン）なし減算及び符号なし加算を用いることで、条件処理を行うことなく上記差分処理及び併合処理をそれぞれ実現できる。

【００３６】

この差分・併合処理による幾つかの演算例を示す。

差分　８２　－　５７　＝　２５、
併合　５７　＋　２５　＝　８２、
差分　２２８　－　３　＝　２２５、
併合　２２５　＋　３　＝　２２８、
差分　５７　－　８２　＝　－２５、　－２５　＋　２５６　＝　２３１、
併合　８２　＋　２３１　＝　３１３、　３１３　－　２５６　＝　５７、
差分　０　－　２５５　＝　－２５５、　－２５５　＋　２５６　＝　１、
併合　２５５　＋　１　＝　２５６、　２５６　－　２５６　＝　０、

差分 ２３ － ２５４ ＝ －２３１、 －２３１ ＋ ２５６ ＝ ２５、
併合 ２５４ ＋ ２５ ＝ ２７９、 ２７９ － ２５６ ＝ ２３。

【００３７】

この差分・併合処理を任意のビット数ｐに適用できるよう一般化すると、
差分：

```
ｄ＝ａ－ｂ;
ｉｆ（ｄ＜０）{
  ｄ＝ｄ＋２＾ｐ;
}。
```

併合：

```
ａ＝ｂ＋ｄ;
ｉｆ（ａ＞＝２＾ｐ）{
  ａ＝ａ－２＾ｐ;
}。
```

となる。ここで２＾ｐは２のｐ乗である。なお、ｐ桁の符号なし演算を用いれば、上記条件処理（ｉｆ以下の処理）は不要となる。

【００３８】

格子点値が符号付きで負の値も取り得る場合は、符号付きの格子点値を符号ビットも含めて正の値と見なすことにより上記の差分処理を適用することができる。併合も処理自体は上記の処理のまま、上記併合処理により得られた値を符号付きの値と見なすだけで対応することができる。

【００３９】

結局ステップＳ１１０からＳ１１６までの処理は次の式で表すことができる。
Ｃ[ｉ:ｊ:ｋ] ← Ｃ[ｉ:ｊ:ｋ] － Ｃ[ｉ:ｊ:ｋ－１]、 ｉ＝０～Ｎ－１、ｊ＝０～Ｍ－１、ｋ＝Ｌ－１～１。

【００４０】

図９に５つの値に対して逆方向に差分処理を行ったときの各値の変化するようすを示す。図６の手順では、ひとつの変換テーブルが順次書き替えられていくが、図９では説明の便宜から更新前と更新後のテーブルを別々に示した。図９のテーブルは、テーブルＣ[ｉ:ｊ:ｋ]において、インデックスｉ及びｊを一定値とし、第３軸（ｋ）方向の格子点値を格子位置の順に並べたものである。一番左のテーブル９０１が初期の状態であり、右にいくに従い最下段の格子点値から順に、そのひとつ前の格子点値との差分に更新されている。処理が終了したテーブル９０５では、一番上の格子点値を除いて、すべてひとつ前の格子点値との差分値となる。なお、逆に図９の右から順に、上の値をひとつ下の値に加算（併合）してゆくと、左端の初期の状態に戻すことができる。

【００４１】

次にステップＳ１０１の詳細な処理手順を図７のフローチャートに示す。図７のステップＳ１２０からＳ１２６では、第２軸の格子点番号０番の格子点値を除く全ての格子点値を、格子点数Ｌ前の格子点値との差分値としている。ステップＳ１１０からＳ１１６と同様にここでも逆方向に処理を進めることにより差分値を上書きしながら処理を行う。この処理は、ＬＵＴを図１のような格子点を三次元的に配置した直方体とした場合に、Ｇ軸（第２軸）方向に隣り合う格子点値の差分を算出することに対応する。図７では、Ｎ、Ｍ、Ｌの値は図６と同様である。変数ｐはＲ（第１軸）方向のインデックスを示す。変数ｑは、変数ｐで与えられるＢＧ面上における格子点位置を示す。そしてｑの初期値はＭ＊Ｌ－１すなわちＧ軸およびＢ軸とも最大値の格子点を示す。その位置からＢ軸に沿って１ずつ着目格子点を移動しつつ、差分を計算する。

【００４２】

ステップＳ１２０からＳ１２６までの処理は次の式で表すことができる。
Ｃ[ｉ:ｊ:ｋ]←Ｃ[ｉ:ｊ:ｋ]－Ｃ[ｉ:ｊ－１:ｋ]、ｉ＝０～Ｎ－１、ｊ＝Ｍ－１～１、ｋ＝０～Ｌ－１。

また、図６の処理を前提とすれば、Ｃ[ｉ:ｊ:ｋ] = Ｃ[ｉ:ｊ:ｋ] － Ｃ[ｉ:ｊ:ｋ－１],
Ｃ[ｉ:ｊ－１:ｋ] = Ｃ[ｉ:ｊ－１:ｋ] － Ｃ[ｉ:ｊ－１:ｋ－１]である。したがって、
C[i:j:k]←(C[i:j:k]-C[i:j:k-1])-(C[i:j-1:k]-C[i:j-1:k-1])
となる。

【００４３】

次にステップＳ１０２の詳細な処理手順を図８のフローチャートに示す。ステップＳ１３０からＳ１３３で第１軸の格子点番号０番の格子点値を除く全ての格子点値を、格子点数のM*Ｌ前の格子点値との差分値とする。この処理はＬＵＴを図１のような格子点を三次元的に配置した直方体とした場合にＲ軸方向に隣り合う格子点値の差分を算出することに対応する。図８では、ＭＮＬの値は図６と同様である。変数ｑは、色変換テーブルの１次元のインデックスを示す。そしてｑの初期値はＮＭＬ－１すなわち図３のテーブルの末尾である。その位置から１ずつ着目格子点を移動しつつ、第１軸方向について隣接する格子点との差分を計算する。

【００４４】

ステップＳ１３０からＳ１３３までの処理は次の式で表すことができる。
Ｃ[ｉ:ｊ:ｋ] ←Ｃ[ｉ:ｊ:ｋ]－Ｃ[ｉ－１:ｊ:ｋ] 、ｉ＝Ｎ－１～１、ｊ＝０～Ｍ－１、ｋ＝０～Ｌ－１。

【００４５】

また、図６および図７の処理を前提とすれば、テーブルＣの全軸差分値Ｃ'は以下のように表せる。

C'[i:j:k] ← ((C[i:j:k]-C[i:j:k-1])-(C[i:j-1:k]-C[i:j-1:k-1]))-((C[i-1:j:k]-C[i-1:j:k-1])-((C[i-1:j-1:k]-C[i-1:j-1:k-1]))
=C[i:j:k]-C[i:j:k-1]-C[i:j-1:k]-C[i-1:j:k]
+C[i:j-1:k-1]+C[i-1:j:k-1]+C[i-1:j-1:k]-C[i-1:j-1:k-1] ...（数式１）。

【００４６】

このように、以上の処理により[０：０：０]以外の全ての格子点値が差分値となる。ただし、数式１において、該当する格子点が存在しない項の値は０とする。この結果、座標[０：０：０]等、マトリクス端部の格子点を含む全ての格子点を数式１（ただし書きも含む）によって表すことができる。数式１により表される値のことを本実施形態では全軸差分と呼ぶ。なお、着目格子点[ｉ：ｊ：ｋ]に関する差分値の求め方は数式１にはかぎられない。上記例では格子点[０：０：０]の値は差分が求められない値である。これは、各軸方向についての差分値は独立して計算することができるためである。すなわち、各軸方向について、着目格子点のマイナス方向にある隣接格子点を差し引くか（これは上記例である。）、それとも着目格子点のプラス方向にある隣接格子点を差し引くかは選択事項である。したがって、格子点[０：０：０]のように減ずる値（すなわち差分計算の第２項）としかならない格子点を差分原点とよぶとすれば、マトリクスの８隅点のどこを差分原点として選んでも良いはずである。したがって、数式１の"ｉ－１"、"ｊ－１"、"ｋ－１"をそれぞれ"ｉ＋１"、"ｊ＋１"、"ｋ＋１"に置換した数式もやはり全軸差分を表す。すべてを置換せずに、一部を置換した場合も同様である。したがって全軸差分C'[i:j:k]の定義を一般化すると以下のようになる。

C'[i:j:k]=C[i:j:k]-C[i:j:k-a3]-C[i:j-a2:k]-C[i-a1:j:k]
+C[i:j-a2:k-a3]+C[i-a1:j:k-a3]+C[i-a1:j-a2:k]-C[i-a1:j-a2:k-a3]...(数式1')。

ただし、ａ１，ａ２，ａ３は、－１または１であり、該当する格子点が存在しない項の値は０とする。

【００４７】

換言すれば、これまでの説明では各軸の差分をとらない格子点値を０番目の格子点値としたが、０番目ではなく各軸の最後の格子点値を、差分をとらない格子点とすることができる。例えば第３軸で最後の格子点値を差分をとらないとした場合には、ステップＳ１１０からＳ１１６のループは０～Ｌ－２の範囲を０から順方向にまわることになる。これは第３軸の逆方向とするために、数式１'におけるパラメータを、ａ１＝－１、ａ２＝１、

ａ３＝１として全軸差分を計算する手順に相当する。

【００４８】

これまで説明した第１の実施形態では、まず第３軸方向の差分処理を行い、続いて第２軸方向、最後に第１軸方向と差分処理を行っている。この順番は任意に変更して処理を実施することが可能であり、３軸方向全ての差分処理が終了した段階では全く同一の結果を得ることができる。このことは、数式１から明らかであろう。すなわち、これまでの説明で各軸方向の差分処理が相互に依存していないこと、差分処理がその順番を変えても同じ結果が得られることから明らかである。

【００４９】

なお、ここまで説明した処理を実施しても変換テーブルのデータ量は全く変化しない。つまり変換テーブルが格納する値を圧縮が効果的に行えるように変換する処理であって、データ量を削減する圧縮処理ではない。だたし、格子点値を３軸方向に差分をとることにより、偏りなく分散していた格子点値が０または２５５の近辺の値に集中するようになる。なぜなら色変換のためのＬＵＴの格子点値は多くの場合滑らかに変化しているため、３軸方向の差分をとることは格子点値を小さくすることにつながるからである。なお、２５５は－１に相当する値である。０付近の値は、２進数で表現すれば比較的長いリーディング０の後に、短い桁の１が表れる数値である。２５５付近の値は、２進数で表現すれば比較的長いリーディング１の後に、短い桁の０が表れる数値である。このような値をエントロピ符号化、たとえばランレングスを利用した圧縮法で可逆圧縮すれば、高圧縮率で圧縮できる。なお各軸の格子位置４０１は、変換テーブルを構成するデータの一部である。したがって、格子位置４０１は全軸差分の処理対象とはならないが、圧縮されたＬＵＴの一部として、あるいは圧縮されたＬＵＴとともに、ＬＵＴを利用するコンピュータに提供される。

【００５０】

以上説明してきた３軸方向全ての差分処理をおのおの全ての出力データに施し、図４のようにそれらを連結して得られたデータに対して、図５Ｂのステップ Ｓ１０７において圧縮処理を実施する。圧縮処理としては完全に元のデータを復元できる手法、いわゆる可逆圧縮あるいはロスレス圧縮、であればどのような手法であっても適応することができる。一例としてランレングス圧縮やＵＳＰ４５５８３０２に開示されているＬＺＷ圧縮として知られる手法などがある。

【００５１】

＜復元処理＞

このように３軸方向に差分処理を行った後、圧縮処理を施したデータを利用して変換処理を実施する処理手順を、図３０Ｂに示す。この処理は、入力された印刷設定に対応した着目色変換テーブルを利用するプリンタドライバが、そのプリンタドライバがインストールされたコンピュータにおいて、着目色変換テーブルを対象として実行する。また、本実施形態では、図３０Ｂの復元プログラムは、図５Ｂのプログラムによって圧縮された色変換テーブルと共にコンピュータにロードされる。ただし、圧縮された色変換テーブルと復元プログラムは別々に同じコンピュータにロードされても良い。

【００５２】

まず、圧縮の復号処理を行う（Ｓ３５０３）。可逆圧縮処理であるので、３軸方向に差分処理を行った後のデータ、すなわち図５Ｂのステップ Ｓ１０７直前のデータを取得することができる。

【００５３】

次にＬＵＴ復元処理を行って、差分処理された変換テーブルを復元する（Ｓ３５０５）。これを出力色空間の各色成分について繰り返す（Ｓ３５０６、Ｓ３５０７）。復元が終了したなら、復元した色変換テーブルを、ＤＲＡＭ１００３あるいはＨＤＤ２００２ａに保存する。保存された色変換テーブルは、色変換のために参照される。ＬＵＴ復元処理は差分処理とほぼ同じ手順である。ただし以下の２点が異なる。第１に、着目格子点値に、そのひとつ前の格子点値を併合（加算）することである。第２は、順方向に併合処理を進

めることである。併合処理を順方向に進めるのは、復号済みのテーブルの格子点値に差分を併合する必要があるためである。

【００５４】

図３０ＡにＬＵＴ復号処理の手順を示す。また、図３３、図３２、図３１に、図３０ＡのステップＳ３５００～Ｓ３５０２それぞれを示す。図３１、図３２、図３３の処理は、それぞれ図６乃至図８の手順のほぼ逆をなぞる処理である。しかし、上記２点が守られれば、着目する軸方向以外の方向に対するテーブルの走査の順序はどのような順序であっても良いので、その順序は必ずしも図６乃至図８それぞれの逆になってはいない。復元処理により、以下の変換がされる。

Ｃ[ｉ:ｊ:ｋ] ← Ｃ[ｉ:ｊ:ｋ] + Ｃ[ｉ:ｊ:ｋ－１]、ｉ＝０～Ｎ－１、ｊ＝０～Ｍ－１、ｋ＝１～Ｌ－１、

Ｃ[ｉ:ｊ:ｋ] ← Ｃ[ｉ:ｊ:ｋ] + Ｃ[ｉ:ｊ－１:ｋ]、ｉ＝０～Ｎ－１、ｊ＝１～Ｍ－１、ｋ＝０～Ｌ－１、

Ｃ[ｉ:ｊ:ｋ] ← Ｃ[ｉ:ｊ:ｋ] + Ｃ[ｉ－１:ｊ:ｋ]、ｉ＝１～Ｎ－１、ｊ＝０～Ｍ－１、ｋ＝０～Ｌ－１。

【００５５】

図３０Ａの例では、図５Ａと逆の順序で復元を行っている。しかし、差分処理を行う順番と、差分処理を復元する順番は無関係に実施することができる。たとえば、Ｂ軸方向、Ｇ軸方向、Ｒ軸方向の順に差分処理を行ったとしても、差分処理の復元をＧ軸、Ｂ軸、Ｒ軸の順のように、順不同で復元処理を行うことができる。なぜなら、差分処理をどのような順序で行っても得られる結果が変わらないのと同じ理由で、差分処理の復元をどのような順序で行っても結果には変化がないからである。この結果、色変換テーブルには元の値が復元される。

【００５６】

以上の手順により本実施形態によれば、色変換テーブルの特徴を利用して、色変換テーブルを高圧縮率で圧縮することができる。また、変換および逆変換によって失われる情報がない。

【００５７】

［第２実施形態］

本実施形態も第１の実施形態と同様に、隣接する格子点値の相関性を活用して圧縮処理の効果を高める手法である。本実施形態では隣接する格子点値の単純な差分を取るのではなく、近隣の格子点値を用いた補間により得られる値と実際の格子点値の差分を取ることにより格子点値を０近辺の値に変換する。本実施形態は、変換精度を向上さるために格子数を増大させたＬＵＴの圧縮に特に有効である。図１０は、本発明の第２実施形態にかかるＬＵＴの変換処理の手順を示すメインフローチャートである。なお、圧縮手順は図５Ｂに示したものと同一である。ただし、ステップＳ１０４が図１０の手順となる。なお本実施形態でも、第１実施形態と同様に出力色成分（ＹＭＣＫ）それぞれの変換テーブルのうち、Ｃ成分の変換テーブルを例として説明するが、全ての色成分について同様に本実施形態の変換処理（図１０）は適用され、その後に圧縮が行われる。

【００５８】

本実施形態は大きく分けてステップＳ２０４の縮小ＬＵＴ作成とステップＳ２０５の差分データ作成から構成される。さらにステップＳ２０４とステップＳ２０５とを、格子点数を減少させながら繰り返し処理を行う。図１０において、まずＮ、Ｍ、Ｌの値をセットする（Ｓ２００～Ｓ２０１）。これらの値の意味は、第１実施形態の図６乃至図８と同様である。そして縮小ＬＵＴが、補間可能な最小のサイズ、すなわち各座標軸の格子点が２となるまでステップＳ２０４において間引き処理を行う。ステップＳ２０５において、縮小ＬＵＴの補間値と、その補間値に対応する、間引き前のＬＵＴの格子点の値との差分データを、各格子点毎に求めて差分テーブルを作成する。図１１は縮小ＬＵＴ作成処理の手順を示すフローチャートである。

【００５９】

図１１では、格子番号が偶数であるか、または最後の格子点である格子点値は縮小ＬＵＴ（縮小後のＬＵＴ）の格子点値とする。この処理を第１軸から第３軸の全ての軸に対して適応することで格子点数をほぼ半減した縮小ＬＵＴを作成する。例えば３軸全ての格子点数が１７のＬＵＴからは各軸について９格子点の縮小ＬＵＴが作成される。縮小はマトリク成分の間引きによって実現される。なお図１１の手順は、変換テーブルを３次元のインデックスで表して実現されている。もちろん第１実施形態のように１次元のインデックスで表しても、第１実施形態で示した変換式に則って、図１１と同様に間引き処理を実現できる。

【００６０】

図１１において、変数ｉ、ｊ、ｋ、ｌ、ｍ、ｎはＤＲＡＭ２００３に確保される。定数Ｎ、Ｍ、ＬもＤＲＡＭに保存される。変数ｉ、ｊ、ｋは、間引き対象となるＬＵＴの入力色空間の第１座標軸（Ｒ）、第２座表軸（Ｇ）、第３座標軸（Ｂ）のインデックスであり、それぞれの格子番号を表す。変数ｌ、ｍ、ｎは、間引き後の縮小ＬＵＴの第１座標軸（Ｒ）、第２座表軸（Ｇ）、第３座標軸（Ｂ）のインデックスであり、それぞれの格子番号を表す。またＣ［ｉ：ｊ：ｋ］は間引き対象のＬＵＴの格子点値を、Ｃｒ［ｌ：ｍ：ｎ］は縮小ＬＵＴの格子点値を示す。図１１の手順では、間引き対象のＬＵＴから、ＲＧＢ各軸について、格子番号が奇数または最大値であるような格子点値を残し、その他を間引いた縮小ＬＵＴとしてＣｒ［ｌ：ｍ：ｎ］が作成される。

【００６１】

図１２は、図１０のステップＳ２０５の差分データを作成する処理の手順を示すフローチャートである。差分データは、元ＬＵＴ（間引き対象ＬＵＴ）の格子点の内、間引かれた格子点の値と、その格子点に対応して縮小ＬＵＴより補間演算により算出された出力データ値との差分である。元ＬＵＴの格子点値の内、各軸のいずれかの格子番号が奇数であり、かつ、最後の格子点ではない格子点の格子点値は、縮小ＬＵＴに含まれていない。そのため、その格子点の格子位置を縮小ＬＵＴの入力データとして出力値を算出する。入力された入力データに相当する格子位置がＬＵＴに登録されていない場合、通常は補間演算により出力値が求められ、出力される。補間手法は既知のどのような手法により行うことも可能であるが、本実施形態のプリンタドライバが実際の色変換処理時に行う補間手法と同じ補間方法を用いるのが最も好ましい。そして、元ＬＵＴにおける間引かれた格子点の格子点値と、補間により縮小ＬＵＴに基づいて算出された出力値との差分が計算され、その値が差分テーブルに格納される。差分テーブルは元ＬＵＴと同じディメンションを持ち、間引かれた格子点値と同じ格子点に、その格子点値と補間値との差分値が登録される。差分計算は第１実施形態で示したものと同じ方法で行う。また補間演算の一例として線形補間を用いる補間演算がある。

【００６２】

図１２において、変数および定数の意味は図１１とほぼ同様である。図１２に特有の変数として、フラグｈｒ、ｈｇ、ｈｂがある。これらのフラグは、その全てが０であった場合、そのときの着目格子点（変数ｉ、ｊ、ｋで格子番号を示される格子点）が間引かれない格子点であることを示す。また（ｒ、ｇ、ｂ）は、輝度値を示す。Ｃｄ［ｉ：ｊ：ｋ］は、着目格子点に係る差分データを示す。図１２の手順では、上述の通り間引かれた格子点に対応する差分データが得られ、差分テーブルの対応する位置に登録される。

【００６３】

間引き前の初期状態において各軸の格子点数が、ＲＧＢそれぞれ５×３×７であるＬＵＴ（図２参照）に対して、図１０の処理を実施した場合を考える。第１段階（第１回目のループ）では、ＲＧＢそれぞれが３×２×４の格子点を持つ縮小ＬＵＴ（図１３に示す）が作成される。そして、残りの８１(＝５*３*７－３*２*４＝１０５－２４)の格子点が間引かれる。これら間引かれる格子点について、縮小ＬＵＴ（図１３）による出力値と、元ＬＵＴ（図２）の格子点値との差分データ（図１４に示す）を作成する。図１４の差分テーブルにおいては、間引かれずに縮小ＬＵＴに含まれる格子点については差分データは不要なので、空欄となっている。また丸印の格子点に差分データが登録される。

【００６４】
例えば初期状態のＬＵＴの３つの各軸の格子位置が図２に示すようである場合は元ＬＵＴの格子点[２:１:４]は、入力データ(１２７、１２８、１９０)に対応する。第２軸の格子番号が奇数なので、この格子点は元ＬＵＴから間引かれる。この入力データ(１２７、１２８、１９０)に対する縮小ＬＵＴ（図１３）の出力値Ｃ'(１２７、１２８、１９０)と、元ＬＵＴ（図２）のＣ[２:１:４]の差分を差分データとする。
【００６５】
第２段階（第２回目のループ）では、第１段階で作成した格子点数３×２×４の縮小ＬＵＴ（図１３）が元ＬＵＴになり、格子点数２×２×３の縮小ＬＵＴ（図１５）が作成される。残りの１２(＝3*2*4－2*2*3＝２４－１２)の格子点は間引かれる。これらの格子点について、格子点数２、２、３の縮小ＬＵＴによる出力と元格子点値との差分データ（図１６）が作成される。作成の仕方は、第１段階と同様である。
【００６６】
第３段階（第３回目のループ）では、第２段階で作成した格子点数２×２×３の縮小ＬＵＴ（図１５）が元ＬＵＴとなり、格子点数２×２×２の縮小ＬＵＴ（図１７）が作成される。残りの４(＝2*2*3－2*2*2＝１２－８)の格子点について、格子点数２、２、２の縮小ＬＵＴによる出力と元の格子点値との差分データ（図１８）が作成される。
【００６７】
縮小ＬＵＴの格子点数が２×２×２となるとこれ以上縮小することができないため処理を終了する（Ｓ２０３）。
【００６８】
図１０の処理により、差分データの多くが０近辺の値となるため、第１実施形態と同様、ランレングス符号化などを用いた可逆圧縮の圧縮率が大きく向上する。なお、これまで説明してきた図１０の処理を実施しても、最終的に得られる縮小ＬＵＴと各段階で作成される差分テーブルとを合計したデータ量は、初期のＬＵＴのデータ量と同じである。つまり本手法も格子点値の変換を目的とした処理であって、データ量を削減する圧縮処理ではない。そしてこの変換後に、変換されたデータの圧縮を行う。
【００６９】
上記処理を１段階のみ実施することでも効果はあるが、縮小ＬＵＴの格子点数が２になるまで繰り返し実施することがより効果的である。１段階のみ実施とは、図１０のループを行わず１回目のステップＳ２０５の終了を以てＬＵＴ変換処理を終了させることである。たとえば、図２のＬＵＴから、図１３の縮小ＬＵＴと図１４の差分テーブルとを生成した段階が１段階の処理に相当する。
【００７０】
また、入力空間の座標軸それぞれ（本例では３軸）の格子点数が異なる場合には、全ての軸について同時に格子点数が２とならない場合もある。この場合、格子点数が２になった軸はそれ以降も格子点数を２に維持し、格子点数が２より大きい軸については前述した処理に従い格子点数を減少させてゆく。全ての軸の格子点数が２になったところで処理を終了する。
【００７１】
なお、圧縮は、図１９のデータ全体を対象として行っても良いし、データ量が小さい上に、あまり効果が上がらないことが予想される格子位置１９０１を除外して残りを圧縮しても良い。
【００７２】
図１０の処理は、変換テーブルが格納する値を、圧縮が効果的に行えるように変換する処理であって、データ量を削減する圧縮処理ではない。ただし、本実施形態においても、格子点値を３軸方向に差分をとることにより、偏りなく分散していた格子点値が０または２５５の近辺の値に集中するようになる。０付近の値は、２進数で表現すれば比較的長いリーディング０の後に、短い桁の１が表れる数値である。２５５付近の値は、２進数で表現すれば比較的長いリーディング１の後に、短い桁の０が表れる数値である。このような

値をエントロピ符号化、たとえばランレングスを利用した圧縮法で可逆圧縮すれば、高圧縮率で圧縮できる。なお各軸の格子位置１９０１は、差分テーブルを構成するデータの一部である。したがって、格子位置１９０１は差分の処理対象とはならないが、圧縮されたＬＵＴの一部として、あるいは圧縮されたＬＵＴとともに、ＬＵＴを利用するコンピュータに提供される。

【００７３】

＜復元処理＞

ＬＵＴの復元には、初期状態のＬＵＴの各軸の格子点数および格子位置が必要であるため、図１９のようにこれらの値（各軸の格子位置１９０１）を差分データ１９０２～１９０４および縮小ＬＵＴ１９０５とともに格納されている必要がある。元ＬＵＴには、図４に示すように各軸の格子位置は与えられているので、それをそのまま保持すればよい。格子点の間引き方を規則的に行う場合は初期状態のＬＵＴの各軸の格子位置のみを格納することによりＬＵＴを復元することができる。なお復元処理は、第１実施形態と同様にたとえばプリンタドライバが、印刷設定を入力された場合に、入力された印刷設定に応じたＬＵＴを選択して、それを復元する。

【００７４】

復元処理は縮小ＬＵＴの格子数、差分データおよび初期状態のＬＵＴの格子数・位置を用いて以下のように行う。元ＬＵＴの格子位置から差分データのデータ数および各差分データの格子点位置を確定できる。なぜなら格子位置の間引き方は規則的であるため、各段階の格子位置を初期状態のＬＵＴの格子位置から決定することができるからである。よって、各段階の縮小ＬＵＴの格子点位置、すなわち差分テーブルの格子点の位置が得られる。こうして、元ＬＵＴの格子位置の情報から縮小ＬＵＴの格子位置と１つ前のＬＵＴの格子位置が得られる。そこで縮小ＬＵＴを使用して、１つ前のＬＵＴの格子位置（すなわち差分テーブルの格子点の位置）を入力とした時の出力値を、格子点値の補間によって得ることができる。その出力値と該当する格子位置の差分データとを併合（加算）することで１段階前のＬＵＴの格子点値を取得することができる。１段階前のＬＵＴとは、間引き処理を行う前のＬＵＴである。この処理を１段階前のＬＵＴのすべての格子点に対して行うことにより、１段階前のＬＵＴを復元することができる。

【００７５】

そしてそれらの処理を、格子点数が初期状態のＬＵＴ（元ＬＵＴ）と同じになるまで繰り返すことによりＬＵＴを完全に復元することができる。

【００７６】

図３４に復元処理手順の例を示す。この例では復元を各格子点毎に行わず、補間により得られたデータのみによる補間値テーブルを作成し、それに対応する差分を加算して１段階上のＬＵＴを復元している。まず、圧縮データを復号する（Ｓ３９０１）。次に、復元に先立って、各段階における縮小ＬＵＴの格子位置を、格子点の間引き方法に基づいて決定し、記憶しておく。そして、まず最低階層の差分テーブルに着目する。これを着目差分テーブルと呼ぶ。なお縮小ＬＵＴとは、ある時点において復元されたＬＵＴを指す。

【００７７】

次に復号したデータ中に含まれる縮小ＬＵＴの値に基づいて、着目差分テーブルの格子点に対応する各格子点の格子点値を補間し、補間テーブルを作成する（Ｓ３９０２）。次に補間テーブルの各格子点値について、着目差分テーブルの対応する格子位置の差分データを加算する（Ｓ３９０３）。次に最上位まで復元したか判定する（Ｓ３９０４）。たとえば復元されたＬＵＴの格子数が、格子位置情報１９０１に含まれる格子位置を一致していれば、最上位まで復元したものと判定できる。復元が終了して入れば、復元したＬＵＴを保存し（Ｓ３９０６）、終了していなければ、現在より１段階上の差分テーブルに着目する（Ｓ３９０５）。もちろん縮小ＬＵＴは、復元されたＬＵＴとする。そしてステップＳ３９０２に戻る。

一方、図２０のように各段階の縮小ＬＵＴの格子位置を順に格納する方法も可能である。先に示したＬＵＴの場合であれば、第３段階の各軸の格子位置であるＲ軸０、２５５、Ｇ

軸０、２５５、Ｂ軸０、２５５を第３段階で作成される縮小ＬＵＴの前に格納する。第２段階の各軸の格子位置であるＲ軸０、２５５、Ｇ軸０、２５５、Ｂ軸０、１９０、２５５を第３段階で作成される差分データの前に格納する。第１段階の各軸の格子位置であるＲ軸０、１２７、２５５、Ｇ軸０、２５５、Ｂ軸０、８５、１９０、２５５を第２段階で作成される差分データの前に格納する。初期状態の格子位置であるＲ軸０、６４、１２７、１９０、２５５、Ｇ軸０、１２８、２５５、Ｂ軸０、４２、８５、１２７、１９０、２３２、２５５を第１段階で作成される差分データの前に格納する。

【００７８】

これらの形式で格子位置を格納すれば、差分処理がどのように行われたかが格子位置情報から判別できる。そのため、差分処理の格子点の間引き方、およびどの格子数まで間引くかという取り決めを変更しても、間引きデータからＬＵＴの復元処理を行うことが可能になる。

【００７９】

以上説明したように本実施形態では、色変換ＬＵＴを、補間値との差分を示す差分テーブルに変換することで、各格子点値を０に近い値に変換することができる。これによって、可逆圧縮した際のＬＵＴの圧縮率を高めることができる。

【００８０】

［第３実施形態］

第３の実施形態では、異なる複数のＬＵＴをマージする。１つの画像処理装置であっても用紙種類や品位、使用インクなどの条件に対応して複数の異なる色変換テーブルを格納していて、条件に従って適切なものを選択して使用している。このような複数の色変換テーブルを保有する装置においては全く同一ではないが類似性の高いＬＵＴが存在することがある。第３の実施形態はその類似性を活用することにより圧縮率を向上させる手法である。

【００８１】

マージはマージ対象とする複数のＬＵＴの格子点値を交互に組み合わせて、新たなＬＵＴ（以後、マージＬＵＴと呼ぶ）を作成する。より具体的には、図２１、図２２、図２３を用いて説明する。図２１に示したＬＵＴａ１０００とＬＵＴｂ１０１０は、それぞれ格子点に対応した格子点値Ｃａ，Ｃｂを格納している。すなわち格子点値こそ互いに独立しているが、その構成（構造）は互いに同一である。マージはＬＵＴ１０２０に示すようにＬＵＴａ１０００の格子点値ＣａとＬＵＴｂ１０１０の格子点値Ｃｂを交互にインターリーブしていく。この処理を各色で行い、マージＬＵＴａｂ１０３０（図２３）を作成する。また、マージＬＵＴａｂ１０３０の先頭にマージ順番を付加する。図２２の例では格子点値Ｃａの次に格子点値Ｃｂを配置しているため、順番はＬＵＴａ→ＬＵＴｂとなる。この順番を示す識別符号１０３０１が、マージＬＵＴａｂ１０３０に付加される。なお図２１～図２３の例では２つのＬＵＴをマージしているが、これに限らず、例えば３つ以上のＬＵＴに用いても良い。また、２格子点以上を交互にマージしてもよい。

【００８２】

図２４は、本発明の第３実施形態にかかる色変換テーブルの変換処理および格納の手順を示すメインフローチャートである。Ｓ１０００からプログラムがスタートし、処理はＳ１００２に移る。Ｓ１００２では、まず全てのＬＵＴを単独で圧縮する（圧縮された単独ＬＵＴを以後、単独圧縮ＬＵＴと呼ぶ）。圧縮は例えばエントロピ符号化などを用いて行われる。圧縮処理が終了すると、処理はＳ１００４に移る。Ｓ１００４では、全てのＬＵＴの組み合わせに対してマージし、圧縮する（圧縮されたマージＬＵＴを以後、圧縮マージＬＵＴと呼ぶ）。本実施形態では全ての組み合わせとしているが、これに限らず、任意のグループを設け、その中で組み合わせてもよい。図２５はＳ１００４の処理の詳細な流れ図である。なお、各単独ＬＵＴに０からの通し番号を付加し、ＬＵＴを一意に特定する。通し番号の割り当て方法はどのようなものでもよく、例えば単独ＬＵＴのデータ量順が挙げられる。また、説明を簡単化するために、２つのＬＵＴをマージする場合を記載する。

【００８３】
Ｓ１１００からスタートし、処理はＳ１１０２に移る。Ｓ１１０２ではＬＵＴを一意に特定する際に使用する変数ｉを初期化する。初期化が終了すると、処理はＳ１１０４に移る。Ｓ１１０４では、変数ｉがＬＵＴの総数Ｅを超えていないかを判別する。超えていない場合、処理はＳ１１０６に移る。Ｓ１１０６ではＬＵＴを一意に特定する際に使用する変数ｊをｉ＋１で初期化する。本処理でのマージ順番は最初にｉ番目のＬＵＴ、次にｊ番目のＬＵＴとしている。マージ順番は圧縮効率には影響がないため、Ｓ１１０６の処理でマージの組み合わせが重複しないようにしている。
【００８４】
初期化が終了すると、処理はＳ１１０８に移る。Ｓ１１０８では、変数ｊがＬＵＴの総数Ｅを超えていないかを判別する。超えていない場合、処理はＳ１１１０に移る。Ｓ１１１０ではｉ番目の単独ＬＵＴとｊ番目の単独ＬＵＴを前述した方法でマージする。マージが終了すると、処理はＳ１１１２に移る。Ｓ１１１２では、Ｓ１１１０で作成したマージＬＵＴを圧縮する。圧縮すると、処理はＳ１１１４に移る。Ｓ１１１４では、変数ｊをインクリメントし、処理はＳ１１０８に戻る。
【００８５】
一方、Ｓ１１０８で変数ｊがＬＵＴの総数Ｅを超えている場合、処理はＳ１１１６に移り、変数ｉをインクリメントし、Ｓ１１０４に戻る。Ｓ１１０４で、変数ｉがＬＵＴの総数Ｅを超えている場合、処理はＳ１１１８に移り、Ｓ１００４は終了する。
【００８６】
圧縮マージＬＵＴの作成が終了すると、処理はＳ１００６に移る。Ｓ１００６では、マージペアリストの作成を行う。本処理では、マージによって圧縮効果の高まる単独ＬＵＴの組み合わせに対してのみマージ処理を行い、そうでない場合は単独ＬＵＴを採用する。そこで、全ての単独ＬＵＴに対して、マージ処理を加えるか、またその相手となる単独ＬＵＴの情報を格納したマージペアリストを作成する。図２６にマージペアリスト１１００の具体例を示す。マージペアリストは単独ＬＵＴの総数Ｅ個の配列となっており、それぞれの配列要素は単独ＬＵＴの通し番号に対応している。配列要素にはマージのペアとなる他の単独ＬＵＴの通し番号（以後、ペア情報と呼ぶ）を格納する。図の例では、０番目の単独ＬＵＴは２番目の単独ＬＵＴとペアとなり、１番目の単独ＬＵＴはペアが存在しない（図中、ＮＡと表記）ことを示している。
【００８７】
図２７はＳ１００８の処理の詳細な流れ図である。なお、各圧縮マージＬＵＴに０からの通し番号を付加し、ＬＵＴを一意に特定する。通し番号の割り当て方法はどのようなものでもよく、例えば圧縮マージＬＵＴのデータ量順が挙げられる。
【００８８】
Ｓ１２００からスタートし、処理はＳ１２０２に移る。Ｓ１２０２からＳ１２１０ではマージによる削減データ量を算出する。まず、Ｓ１２０２では、圧縮マージＬＵＴを一意に特定する際に使用する変数ｍを初期化し、Ｓ１００４で作成した圧縮マージＬＵＴの総数Ｍを単独ＬＵＴの総数Ｅを用いて算出する。初期化が終了すると、処理はＳ１２０４に移る。Ｓ１２０４では、変数ｍが圧縮マージＬＵＴの総数Ｍを超えていないかを判別する。超えていない場合、処理はＳ１２０６に移る。Ｓ１２０６では以降の処理対象となる圧縮マージＬＵＴを変数ｍで特定する。対象ＬＵＴが決定すると、処理はＳ１２０８に移る。
【００８９】
Ｓ１２０８では、対象ＬＵＴを構成する２つの単独圧縮ＬＵＴのデータ量から対象ＬＵＴのデータ量を減算し、対象ＬＵＴの削減データ量を算出する。削減データ量を算出すると、処理はＳ１２０９に移る。Ｓ１２０９では対象ＬＵＴを削減データ量リストに登録する。削減データ量リストは、圧縮マージＬＵＴの通し番号と削減データ量によって構成された配列である。全ての圧縮マージＬＵＴを削減データ量リストに登録すると、処理はＳ１２１０に移り、変数ｍをインクリメントして、Ｓ１２０４に戻る。

【００９０】
Ｓ１２０４で、変数ｍが圧縮マージＬＵＴの総数Ｍを超えている場合、処理はＳ１２１２に移る。Ｓ１２１２では、Ｓ１２０９で作成した削減データ量リストを削減データ量が多い順に並べ直す。削減データ量リストの並べ直しが終了すると、処理はＳ１２１４、Ｓ１２１６に移る。
【００９１】
Ｓ１２１４では、削減データ量リストに登録されている圧縮マージＬＵＴを一意に特定する際に使用する変数ｉ、Ｓ１２１６ではペアリストを「ＮＡ」で初期化する。初期化が終了すると、処理はＳ１２１８に移る。Ｓ１２１８では、変数ｉが圧縮マージＬＵＴの総数Ｍを超えていない、かつ削減データ量がゼロより大きいかを判別する。条件を満たす場合、処理はＳ１２２０に移る。Ｓ１２２０では削減データ量リストにおけるｉ番目の圧縮マージＬＵＴを以降の処理対象とする。対象圧縮マージＬＵＴが決定すると、処理はＳ１２２２に移る。Ｓ１２２２では対象圧縮マージＬＵＴを構成する単独ＬＵＴが他のマージに使用さていないかを判別する。図２６の例において、１番目と３番目の単独ＬＵＴから構成された圧縮マージＬＵＴが対象の場合、共にマージのペアは存在していない（図中、ＮＡと表記）。しかし、１番目と２番目の単独ＬＵＴから構成された圧縮マージＬＵＴが対象の場合、２番目の単独ＬＵＴは既に他の単独ＬＵＴとペアが決定している。片方でもペアが既に決定している場合、処理はＳ１２２６に移る。
【００９２】
一方、決定していない場合、Ｓ１２２４に移る。Ｓ１２２４では、マージペアリストにペアとなる単独ＬＵＴの情報（ペア情報と呼ぶ。）を格納する。ペア情報を格納すると、処理はＳ１２２６に移り、変数ｉをインクリメントして、処理はＳ１２１８に戻る。Ｓ１２１８で、前述した条件を満たさない場合、処理はＳ１２２８に移り、Ｓ１００６は終了する。
【００９３】
Ｓ１００６が終了すると、処理はＳ１００８に移る。Ｓ１００８では、Ｓ１００６で作成したマージペアリストをもとに、Ｓ１００２で作成した単独圧縮ＬＵＴおよびＳ１００４で作成した圧縮マージＬＵＴをメモリーに格納する。図２８はその具体例であり、図２６のマージペアリストをもとにしている。ＬＵＴ格納領域１１２２には、マージペアリストで指定されているマージペアに対応した圧縮ＬＵＴをメモリーに格納する。図中、ＭＬ．ｔｂｌは圧縮マージＬＵＴ、ＳＬ．ｔｂｌは単独圧縮ＬＵＴを表し、添え字は通し番号に対応している。インデックス領域１１２０は単独ＬＵＴの総数Ｅの配列となっており、それぞれの配列は単独ＬＵＴの通し番号に対応している。配列には上記で格納した圧縮ＬＵＴが格納されているアドレス番地を格納する。図の例では、０番目と２番目の単独ＬＵＴはマージのペアであるため、それらに対応した圧縮マージＬＵＴが格納されているアドレス番地（図中、Ａ０で表記）をそれぞれに格納する。圧縮ＬＵＴの格納が終了すると、処理はＳ１０１０に移り、本処理は終了する。インデックス領域１１２０は、圧縮において用いられる単独圧縮ＬＵＴまたは圧縮マージＬＵＴを示すリストである。本実施形態ではポインタのリストであるが、もちろんテーブルそのもののリストであっても構わない。本実施形態では、ポインタで実現されている場合も含めて、テーブルのリストと呼ぶことにする。
【００９４】
また、メモリーに格納した圧縮ＬＵＴから特定の圧縮ＬＵＴを取得するには、まずインデックス領域１１２０から圧縮ＬＵＴが格納されているアドレス番地を取得する。次に、そのアドレス番地にある圧縮ＬＵＴを解凍する。そのＬＵＴがマージＬＵＴである場合には、マージＬＵＴに格納しているマージ順番情報をもとに、単独ＬＵＴに分離する。
【００９５】
上記処理を行う事によって、特に単独ＬＵＴでは連続していない格子点値であっても、類似性の高い単独ＬＵＴをマージすることによって、同じ値が連続する可能性を飛躍的に向上させえる。これによって、圧縮効率をより高めることができる。

【００９６】
＜復元処理＞
本手法により変換されたＬＵＴを復元するには、図２８のリスト１１２０，１１２２に対して、図２４の逆の処理を実行すればよい。すなわち、ＭＰＵは、図２８の圧縮マージテーブルと単独圧縮テーブルとのリストをインデックス０から読み込む。そして、そのリストが示す圧縮テーブル（圧縮マージテーブルと単独圧縮テーブルとを圧縮テーブルと総称する。）を読み、単独圧縮テーブルであれば、その解凍を行う。一方、圧縮マージテーブルであれば、解凍した後、マージ順を示す識別符号に従って、マージされているテーブルを、個別のテーブルに分解する。このようにして復元された色変換テーブルが、画像形成に使用される。
【００９７】
本実施形態では、複数のテーブルがマージされた圧縮マージテーブルと、単独圧縮テーブルとのうち、圧縮率が高いものを選択することができる。このため、圧縮率の向上に寄与する。
【００９８】
さて以上をまとめると以下のようになる。本実施形態は、色変換テーブルを圧縮して記憶媒体に格納するための色変換テーブル圧縮方法を開示する。そして互いに同じ構成を持つ少なくとも２つの色変換テーブルに格納されているデータを互いにマージしてマージテーブルとし、マージテーブルに格納されているマージデータを圧縮して圧縮マージテーブルとする。そしてマージデータが色変換テーブルのデータをマージしていることを示す情報と前記マージテーブルとを記録媒体に格納する格納ステップとを有している。
【００９９】
さらに、色変換テーブルに格納されているデータを単独で圧縮して単独圧縮色変換テーブルとし、圧縮マージテーブルと、圧縮マージテーブルを構成する色変換テーブルの単独圧縮色変換テーブルとのうち、データ量の少ない方を選択する。そして選択したテーブルを記録媒体に格納する。これにより単独圧縮とマージ圧縮のうち効率の高い方を利用できる。
【０１００】
さらに、圧縮テーブル選択ステップは、圧縮マージテーブルのデータ量と、圧縮マージテーブルを構成する色変換テーブルを単独で圧縮して得られたデータ量との差分値である削減データ量を算出する。そして削減データ量の順に圧縮マージテーブルを並べた削減データ量リストを作成する。そして削減データ量リストから削減データ量の大きい順に、削減データ量が０より大きい圧縮マージテーブルを選択する。また削減データ量リストにおいて削減データ量が０以下の圧縮マージテーブルについては、当該圧縮マージテーブルを構成する色変換テーブルの単独圧縮テーブルを選択する。こうして、より高効率で色変換テーブルを圧縮できる。
【０１０１】
［第４実施形態］
第４の実施形態は第１の実施形態と第２の実施形態の手法を効果的に組み合わせたものである。第２の手法で縮小ＬＵＴおよび差分テーブルを作成し、格子点数が２になる以前であってもその処理を停止する。そうして残った格子点数が２以上の縮小ＬＵＴに対して第１の実施形態の手法で全軸方向差分をとる。こうして得られた、全軸方向差分化された縮小ＬＵＴと差分テーブルとを圧縮する。
【０１０２】
あるいは、縮小ＬＵＴに対して全軸方向差分をとり、差分データとともに圧縮したデータ量と、縮小ＬＵＴをさらに一段階縮小して、その差分データと縮小ＬＵＴの全軸方向差分をとったＬＵＴを圧縮したデータ量とを比較する。そして、縮小を進めた場合にデータ量が低減されない場合には、ＬＵＴをさらに一段階縮小する前の状態で、全軸方向差分化された縮小ＬＵＴと差分テーブルとを圧縮する。低減された場合には、上記処理を繰り返す、データ量がより小さくなる限り繰り返す。こうすることで、第１の手法と第２の手法

の最適な組み合わせを得ることができる。以上の手順によりより圧縮率を向上させることができる。

【０１０３】

［第５実施形態］

第５の実施形態は、第１の実施形態と第３の実施形態の手法を効果的に組み合わせたものである。本実施形態では、第３の実施形態の手法を３軸方向差分処理を行ったＬＵＴに対して処理を行う。すなわち、まず、全てのＬＵＴに対して第１の実施形態で示した３軸方向差分処理を行う。すなわち図５Ｂのステップ Ｓ１０４まで実行する。そして３軸方向差分が施されたＬＵＴに対して第３の実施形態に示したＬＵＴ間マージを行う。すなわち、得られた全軸差分テーブルを対象として、図２４の処理を実行する。

【０１０４】

復号も、第１実施形態と第３実施形態と組み合わせればよい。すなわちマージされ圧縮された全軸差分テーブルを解凍してマージ以前の個別の全軸差分テーブルに戻す。これを第１実施形態と同じ要領で通常の色変換テーブルに戻せばよい。

【０１０５】

第１の実施形態の説明で示した通り、第１の手法を実施してもＬＵＴのデータ量は変化せず、格子点値が差分された値に変化するだけである。また第３の実施形態はＬＵＴの格子点値の内容に全く関係なく実施可能である。そのため、第１の実施形態のＬＵＴ変換方法と、第３の実施形態のＬＵＴ変換方法とを全く変更することなく適応することができる。すなわち、３軸方向差分処理を行うことにより各ＬＵＴの特徴が低減され、複数のＬＵＴ間の類似性が高まる。そのため第３の実施形態による手法がより効果的となる。

【０１０６】

これをまとめると、本実施形態の色変換テーブルを圧縮して記憶媒体に格納するための色変換テーブル圧縮方法は、以下の通りである。互いに同じ構成を持つ少なくとも２つの色変換テーブルに格納されているデータを、当該色変換テーブルの入力空間を規定する複数の軸それぞれの方向について、隣接する格子間の差分値にそれぞれ変換して全軸差分テーブルとする全軸変換ステップを持つ。本実施形態では少なくとも２つの色変換テーブルはＲＧＢ各色成分の色変換テーブルである。もちろん、色成分が２つであれば、「少なくとも２つ」は２つと解釈されるし、４つ以上であれば「少なくとも２つ」は４つと解釈される。

【０１０７】

さらに、その少なくとも２つの全軸差分テーブルに格納されているデータを互いにマージしてマージ全軸差分テーブルとする。当該マージ全軸差分テーブルに格納されているマージデータを圧縮して圧縮マージ全軸差分テーブルとする。さらに、当該マージデータが前記全軸差分テーブルのデータをマージしていることを示す情報と当該圧縮マージ全軸差分テーブルを記録媒体に格納する。

【０１０８】

加えて、前記全軸差分テーブルに格納されているデータを単独で圧縮し単独圧縮全軸差分テーブルとし、圧縮マージ全軸差分テーブルと前記圧縮マージ全軸差分テーブルを構成する単独圧縮全軸差分テーブルのうち、データ量の少ない方を選択する。その選択した圧縮テーブルのみ記録媒体に格納する。

【０１０９】

また、圧縮テーブルを選択するステップでは、前記圧縮マージ全軸差分テーブルのデータ量と、前記圧縮マージ全軸差分テーブルを構成する全軸差分テーブルを単独で圧縮して得られたデータ量との差分値である削減データ量を算出する。そして、削減データ量の順に圧縮マージ全軸差分テーブルを並べたリストを作成し、削減データ量リストから削減データ量の大きい順に、削減データ量が０より大きい圧縮マージ全軸差分テーブルを選択する。一方その削減データ量リストにおいて削減データ量が０以下の圧縮マージ全軸差分テーブルについて当該圧縮マージ全軸差分テーブルを構成する全軸差分テーブルの圧縮テーブルを選択する。

【０１１０】
このような手順により、高効率で色変換テーブルを圧縮することができる。
【０１１１】
［第６実施形態］
第６の実施形態は第２の実施形態と第３の実施形態の手法を効果的に組み合わせたものである。まず全てのＬＵＴに対して第２の手法による変換（間引きおよび全軸差分）を施し、ＬＵＴを縮小ＬＵＴと差分データにする。第２の手法により縮小ＬＵＴと差分データに変換されたＬＵＴデータの間で第３の手法で示した手順でマージにする。すなわち図１０の処理を実行する。第２の手法が施されてもデータ量は変化しないため、第２の手法を施す前のＬＵＴに対するのと全く同様な方法で第３の手法を適応することができる。すなわち、図１９あるいは図２０に示す複数のテーブルをマージする。そして、図２８のリストを作成する。
【０１１２】
解凍は、圧縮の逆の処理を行う。すなわち、圧縮されたテーブルを解凍し、マージされていれば元に戻す。その後は第２実施形態と同じ要領で復元できる。
【０１１３】
なお、第３の実施形態の変換は、第２の実施形態の変換によってえられた縮小ＬＵＴのみならず、差分テーブルについて適用することもできる。
【０１１４】
以上の手順をまとめると以下のようになる。すなわち、本実施形態は、色変換テーブルを圧縮して記憶媒体に格納するための色変換テーブル圧縮方法である。そして、互いに同じ構成を持つ少なくとも２つの第１の色変換テーブルを構成する要素を間引いて、データ量の少ない第２の色変換テーブルをそれぞれ作成する間引きステップを有する。ここで第１及び第２の色変換テーブルは、本実施形態ではＲＧＢの各色成分について与えられる。
【０１１５】
次に、少なくとも２つの第１の色変換テーブルの要素のうち第２の色変換テーブルに含まれない要素の値を、少なくとも２つの第１の色変換テーブルの要素と第２の色変換テーブルを補間して得た要素値との差分値にそれぞれ変換する差分ステップを有する。ここで少なくとも２つとは、本実施形態においてはＲＧＢ各色成分の色変換テーブルである。これにより間引かれた色変換テーブルについて差分テーブルが作成される。
【０１１６】
次に、ＲＧＢの各色について、第２の色変換テーブルと差分値とをそれぞれ連結して連結テーブルとする。これは本実施形態においては、図１９のテーブルに相当する。そして、少なくとも２つの連結テーブルに格納されているデータを互いにマージしてマージ連結テーブルとする。当該マージ連結テーブルに格納されているマージデータを圧縮して圧縮マージ連結テーブルとする。そして当該マージデータが少なくとも２つの連結テーブルのデータを互いにマージしていることを示す情報と、圧縮マージ連結テーブルとを記録媒体に格納する。
【０１１７】
さらに以下のステップを付加することもできる。前記連結テーブルに格納されているデータを単独で圧縮して単独圧縮連結テーブルとし、圧縮マージ連結テーブルと圧縮マージ連結テーブルを構成する単独圧縮連結テーブルのうち、データ量の少ない方を選択する。そして選択した圧縮テーブルを記録媒体に格納する。
【０１１８】
さらに、以下のステップを付加することもできる。圧縮テーブルを選択するステップでは、圧縮マージ連結テーブルのデータ量と、圧縮マージ連結テーブルを構成する連結テーブルを単独で圧縮して得られたデータ量との差分値である削減データ量を算出する。そして削減データ量の順に前記マージ連結テーブルを並べたリストを作成する。削減データ量リストから削減データ量の大きい順に、削減データ量が０より大きい圧縮マージ連結テーブルを選出する。削減データ量リストにおいて削減データ量が０以下の当該マージ連結テ

ーブルについて当該圧縮マージ連結テーブルを構成する連結テーブルの圧縮テーブルを選択する。

【０１１９】

以上の手順により一層圧縮率を向上させることが可能となる。

【０１２０】

なお本発明は、複数の機器（例えばホストコンピュータ、インタフェイス機器、リーダ、プリンタなど）から構成されるシステムに適用しても、一つの機器からなる装置（例えば、複写機、ファクシミリ装置など）に適用してもよい。また本発明の目的は、前述の実施形態の機能を実現するプログラムコードを記録した記録媒体を、システムあるいは装置に供給し、そのシステムあるいは装置のコンピュータが記憶媒体に格納されたプログラムコードを読み出し実行することによっても達成される。この場合、記憶媒体から読み出されたプログラムコード自体が前述した実施形態の機能を実現することになり、そのプログラムコード自体およびプログラムコードを記憶した記憶媒体は本発明を構成することになる。

【０１２１】

また、本発明には、プログラムコードの指示に基づき、コンピュータ上で稼働しているオペレーティングシステム(OS)などが実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれる。さらに、記憶媒体から読み出されたプログラムコードが、コンピュータに挿入された機能拡張カードやコンピュータに接続された機能拡張ユニットに備わるメモリに書込まれた場合についても、本発明は適用される。その場合、書き込まれたプログラムコードの指示に基づき、その機能拡張カードや機能拡張ユニットに備わるCPUなどが実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される。

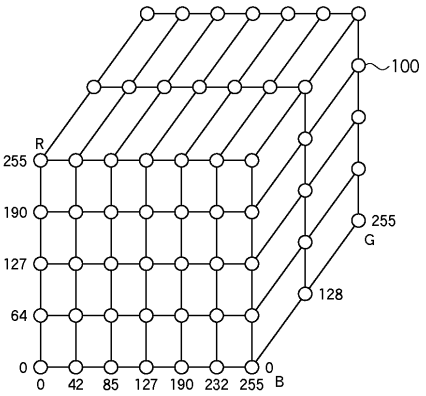
【図面の簡単な説明】

【０１２２】

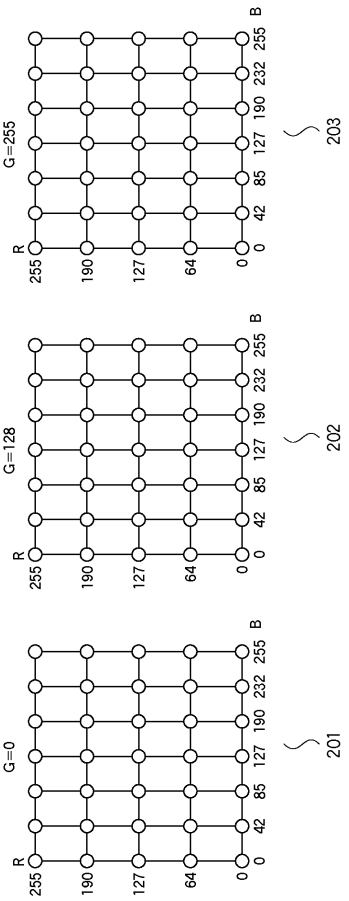
【図１】３軸ＬＵＴを模式的に示した図
【図２】３軸ＬＵＴを２次元的に展開した図
【図３】３軸ＬＵＴを１次元テーブルに展開した図
【図４】複数の出力データを格納する３軸ＬＵＴを１次元テーブルに展開した図
【図５Ａ】第１実施形態のメインフローチャート
【図５Ｂ】第１実施形態のメインフローチャート
【図６】第１実施形態の部分処理のフローチャート
【図７】第１実施形態の部分処理のフローチャート
【図８】第１実施形態の部分処理のフローチャート
【図９】格子点値の差分処理による変化
【図１０】第２実施形態のメインフローチャート
【図１１】２実施形態の部分処理のフローチャート
【図１２】第２実施形態の部分処理のフローチャート
【図１３】第２実施形態の第１段階の縮小ＬＵＴ
【図１４】第２実施形態の第１段階の差分データ
【図１５】第２実施形態の第２段階の縮小ＬＵＴ
【図１６】第２実施形態の第２段階の差分データ
【図１７】第２実施形態の第３段階の縮小ＬＵＴ
【図１８】第２実施形態の第３段階の差分データ
【図１９】第２実施形態の圧縮対象となるデータ
【図２０】第２実施形態の圧縮対象となるデータ
【図２１】２つの３軸ＬＵＴを１次元テーブルに展開した図
【図２２】２つの３軸ＬＵＴをマージして１次元テーブルに展開した図
【図２３】２つの３軸ＬＵＴを各色でマージして１次元テーブルに展開した図
【図２４】第３実施形態のメインフローチャート

【図２５】圧縮マージLUT作成手順を示すフローチャート
【図２６】マージペアリスト
【図２７】マージペアリスト作成手順を示すフローチャート
【図２８】圧縮LUTのメモリー格納形態を示す図
【図２９】コンピュータのブロック図
【図３０Ａ】第１実施形態におけるＬＵＴの復元手順のメインフローチャート
【図３０Ｂ】ＬＵＴの復号手順のメインフローチャート
【図３１】第１実施形態の復元処理の部分フローチャート
【図３２】第１実施形態の復元処理の部分フローチャート
【図３３】第１実施形態の復元処理の部分フローチャート
【図３４】第２実施形態の復元処理のフローチャート
【図３５】第３実施形態の復元処理のフローチャート

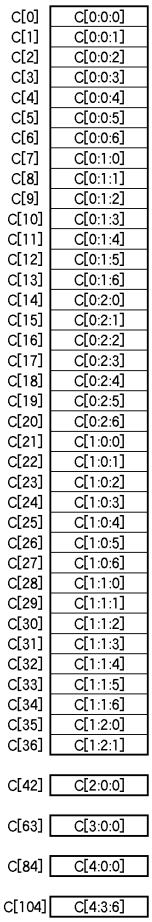
【図１】



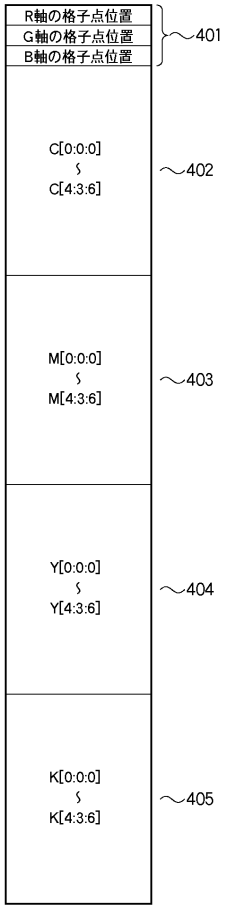
【図２】



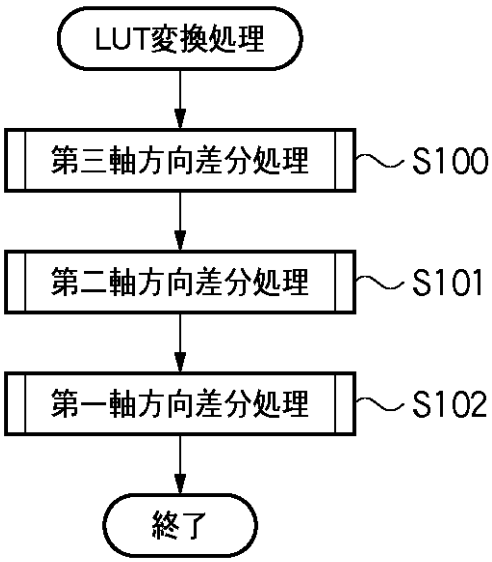
【図３】


C[0] C[0:0:0]
C[1] C[0:0:1]
C[2] C[0:0:2]
C[3] C[0:0:3]
C[4] C[0:0:4]
C[5] C[0:0:5]
C[6] C[0:0:6]
C[7] C[0:1:0]
C[8] C[0:1:1]
C[9] C[0:1:2]
C[10] C[0:1:3]
C[11] C[0:1:4]
C[12] C[0:1:5]
C[13] C[0:1:6]
C[14] C[0:2:0]
C[15] C[0:2:1]
C[16] C[0:2:2]
C[17] C[0:2:3]
C[18] C[0:2:4]
C[19] C[0:2:5]
C[20] C[0:2:6]
C[21] C[1:0:0]
C[22] C[1:0:1]
C[23] C[1:0:2]
C[24] C[1:0:3]
C[25] C[1:0:4]
C[26] C[1:0:5]
C[27] C[1:0:6]
C[28] C[1:1:0]
C[29] C[1:1:1]
C[30] C[1:1:2]
C[31] C[1:1:3]
C[32] C[1:1:4]
C[33] C[1:1:5]
C[34] C[1:1:6]
C[35] C[1:2:0]
C[36] C[1:2:1]
C[42] C[2:0:0]
C[63] C[3:0:0]
C[84] C[4:0:0]
C[104] C[4:3:6]

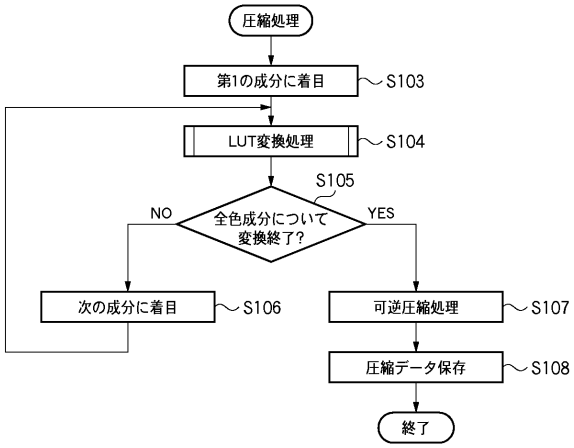
【図４】


R軸の格子点位置
G軸の格子点位置
B軸の格子点位置
401
C[0:0:0]
〜
C[4:3:6]
402
M[0:0:0]
〜
M[4:3:6]
403
Y[0:0:0]
〜
Y[4:3:6]
404
K[0:0:0]
〜
K[4:3:6]
405

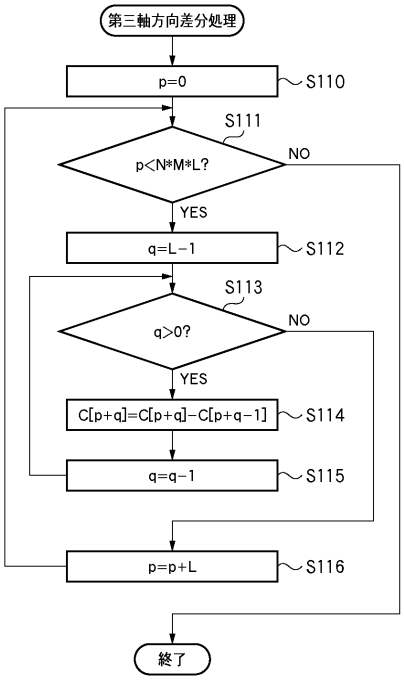
【図５Ａ】


LUT変換処理
第三軸方向差分処理
S100
第二軸方向差分処理
S101
第一軸方向差分処理
S102
終了

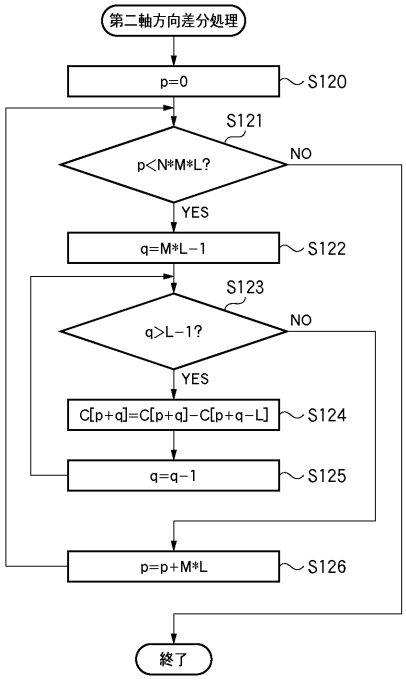
【図５Ｂ】


圧縮処理
第1の成分に着目
S103
LUT変換処理
S104
S105
全色成分について
変換終了?
NO
YES
次の成分に着目
S106
可逆圧縮処理
S107
圧縮データ保存
S108
終了

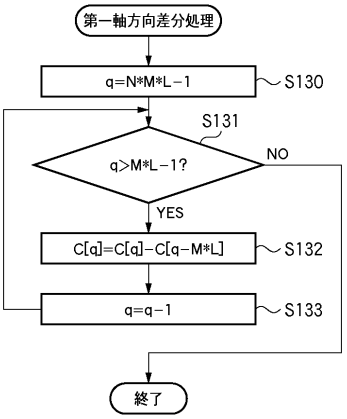
【図６】



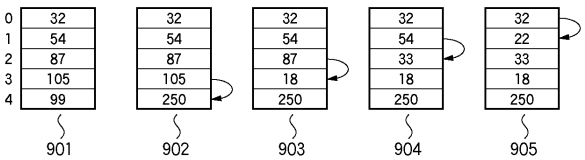
【図７】



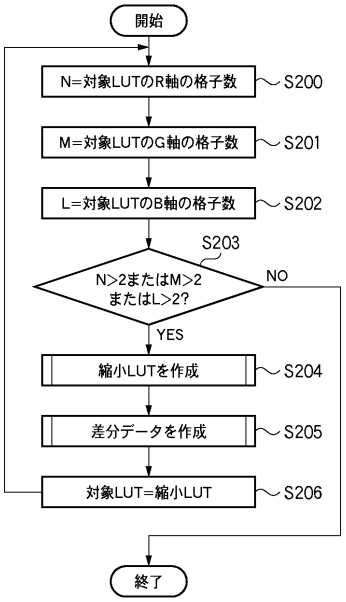
【図８】



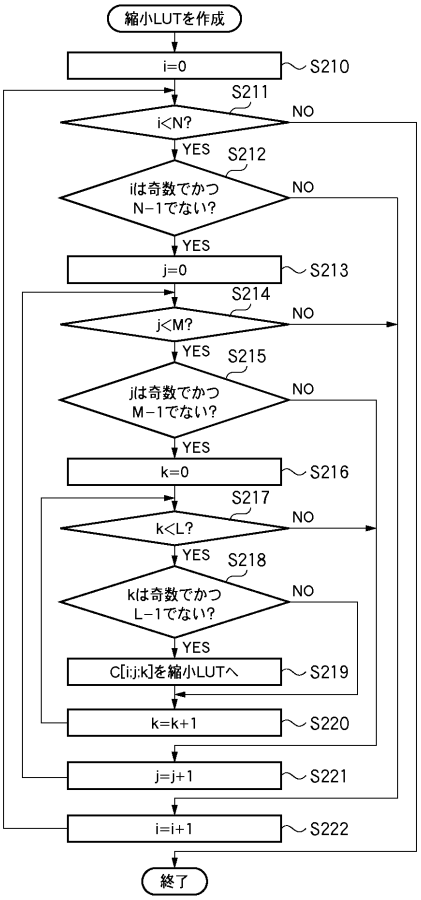
【図９】



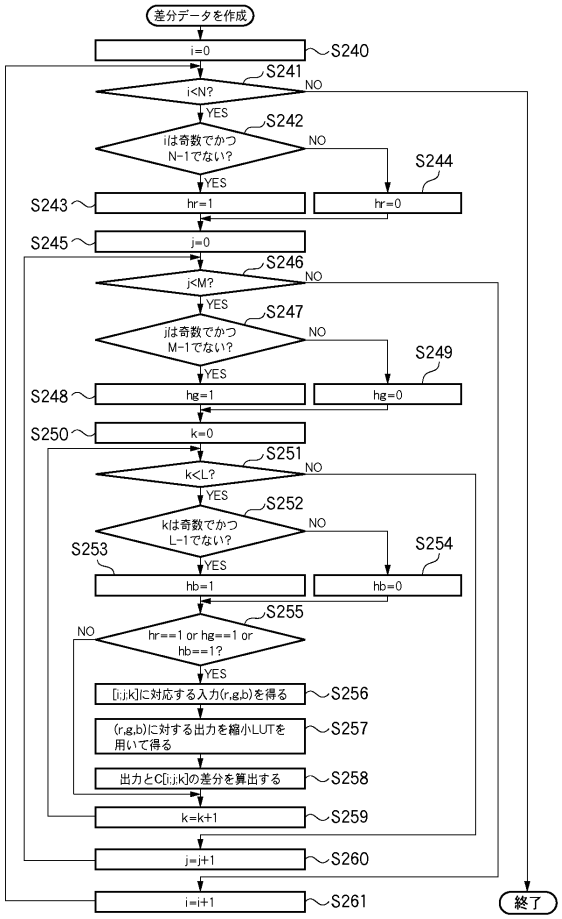
【図１０】



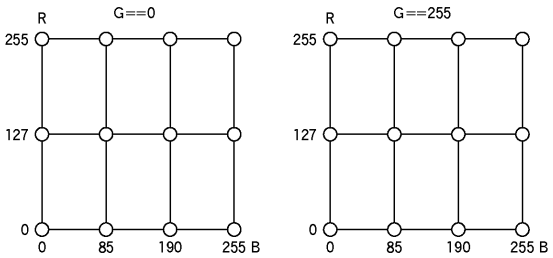
【図１１】



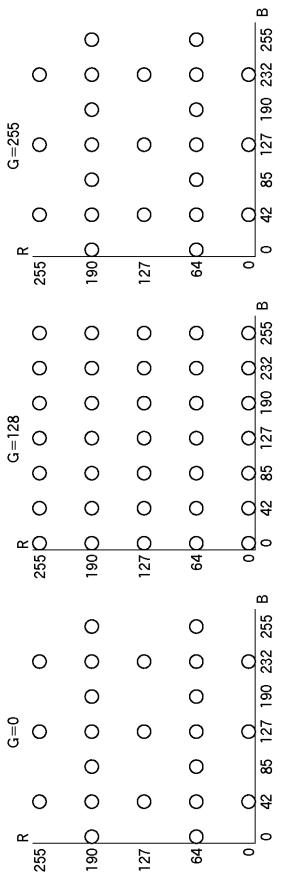
【図１２】



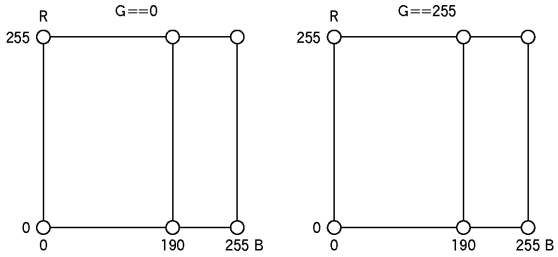
【図１３】



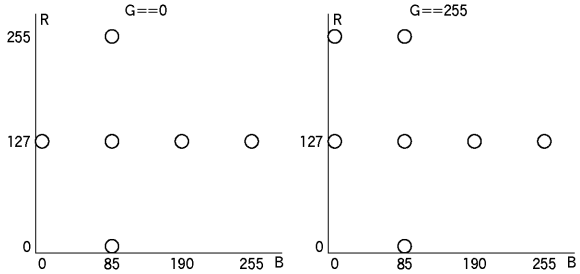
【図１４】



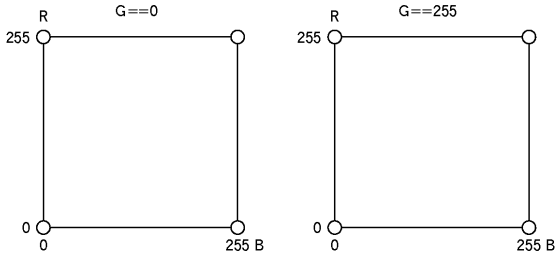
【図１５】



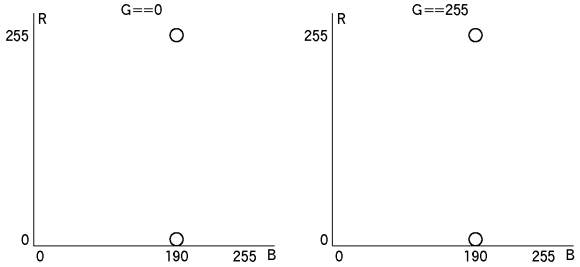
【図１６】



【図１７】

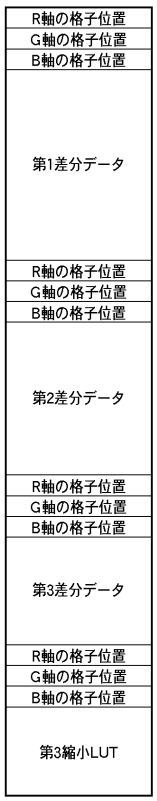


【図１８】

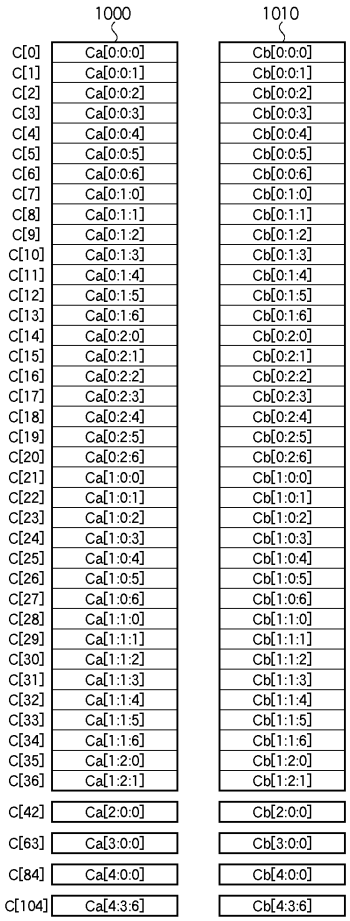


【図１９】

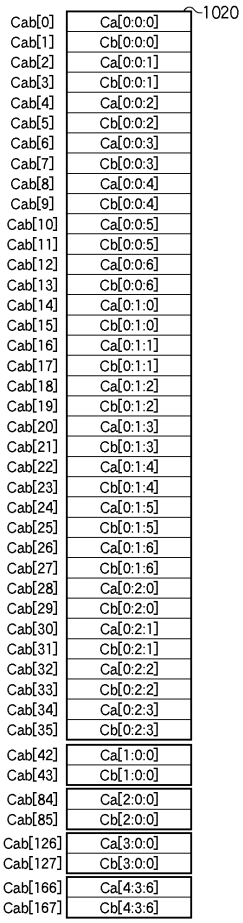
【図２０】



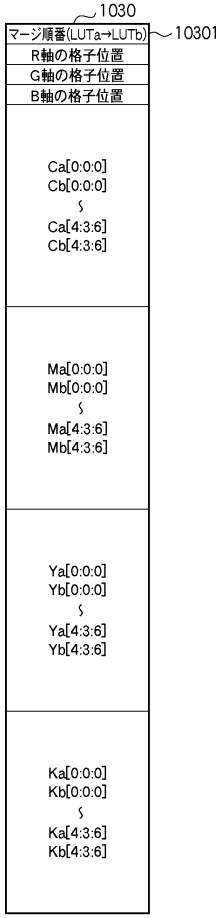
【図２１】



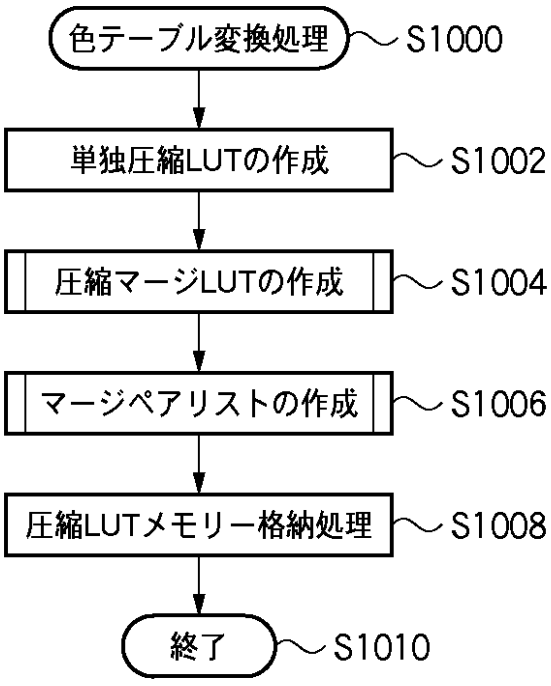
【図２２】



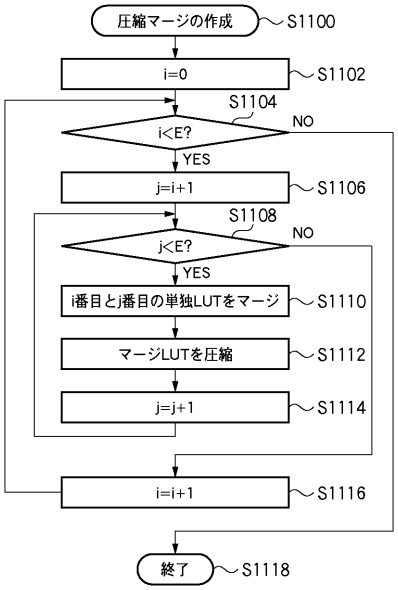
【図２３】



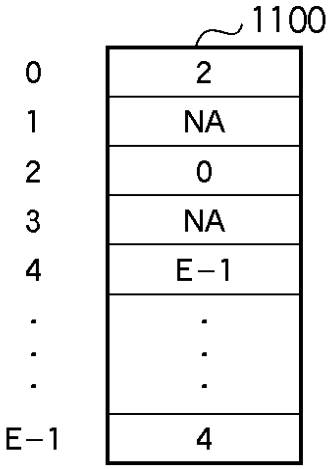
【図２４】



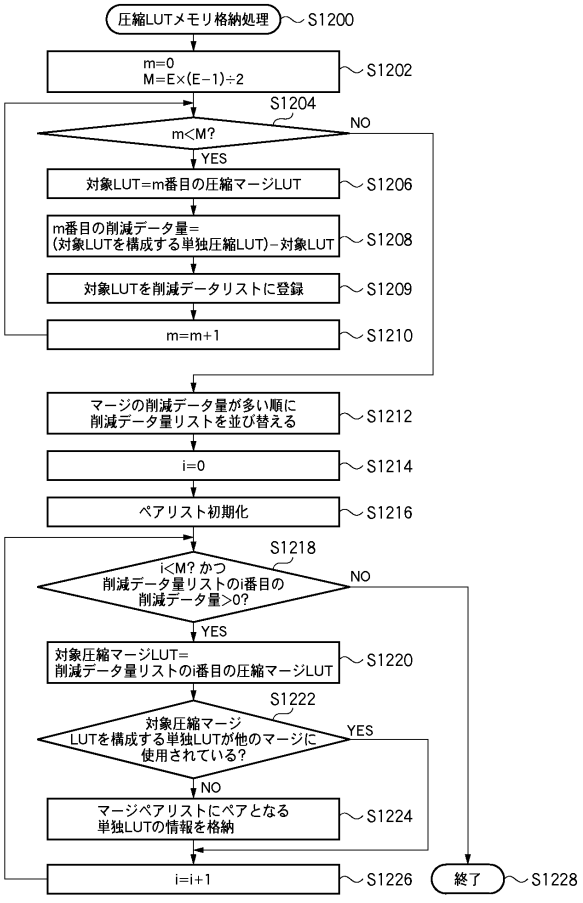
【図２５】



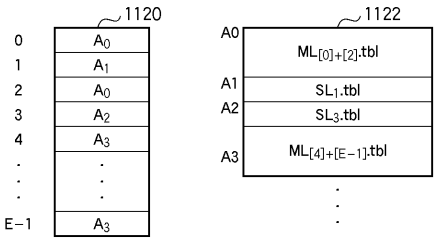
【図２６】



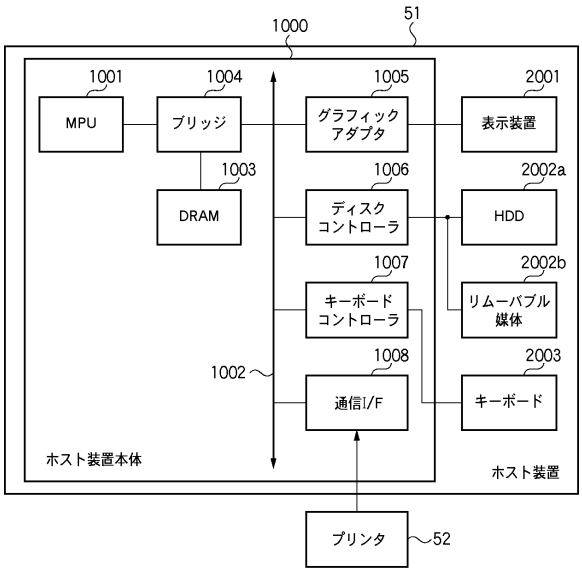
【図２７】



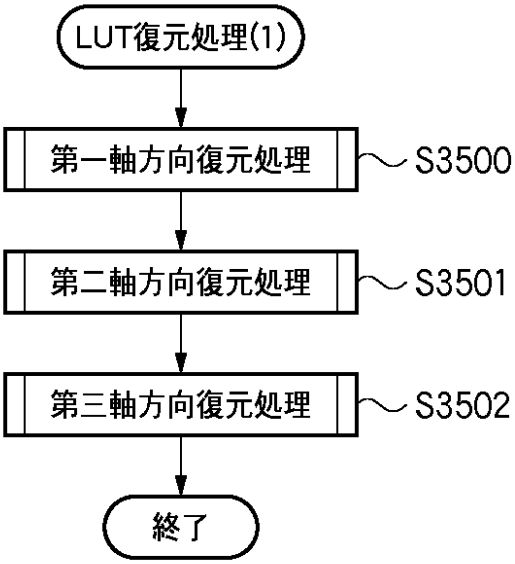
【図２８】



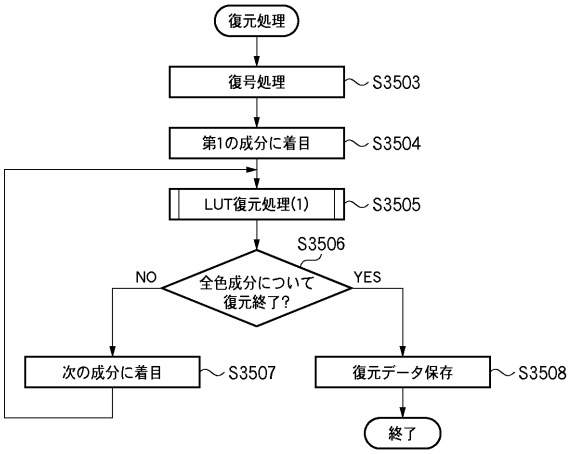
【図２９】



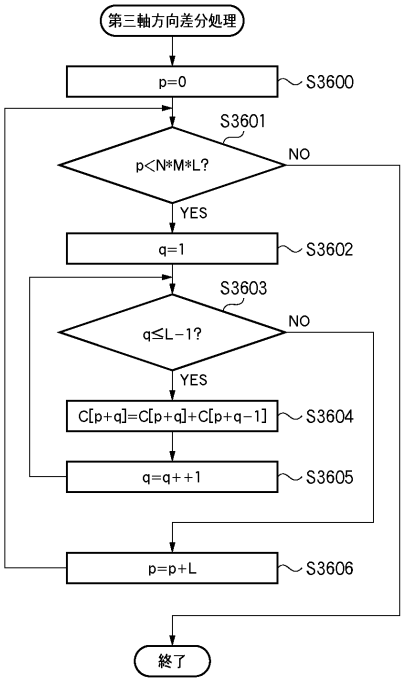
【図３０Ａ】



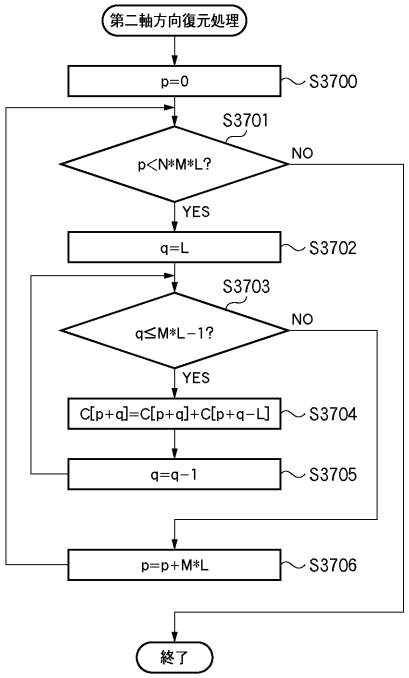
【図３０Ｂ】



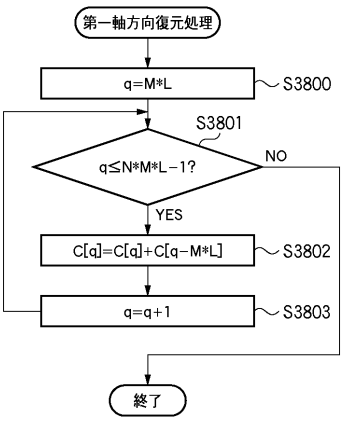
【図３１】



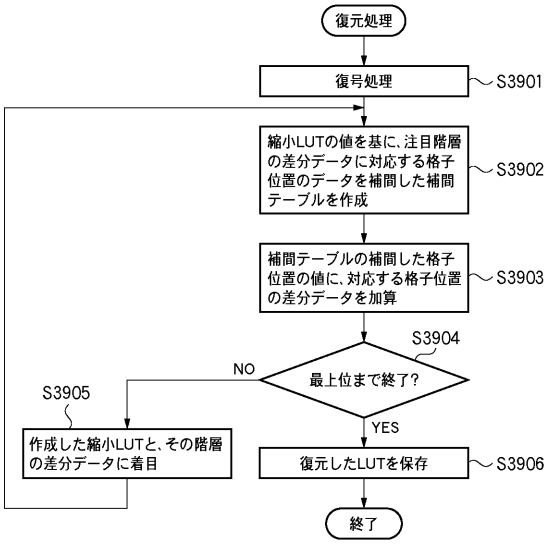
【図３２】



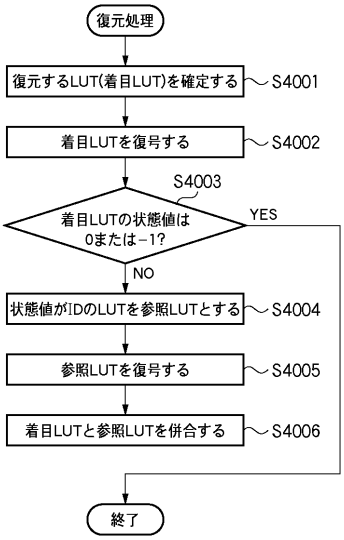
【図３３】


第一軸方向復元処理
q=M*L
S3800
S3801
q≦N*M*L−1?
NO
YES
C[q]=C[q]+C[q−M*L]
S3802
q=q+1
S3803
終了

【図３４】


復元処理
復号処理
S3901
縮小LUTの値を基に、注目階層の差分データに対応する格子位置のデータを補間した補間テーブルを作成
S3902
補間テーブルの補間した格子位置の値に、対応する格子位置の差分データを加算
S3903
S3904
最上位まで終了?
NO
YES
S3905
作成した縮小LUTと、その階層の差分データに着目
復元したLUTを保存
S3906
終了

【図３５】


復元処理
復元するLUT(着目LUT)を確定する
S4001
着目LUTを復号する
S4002
S4003
着目LUTの状態値は
0または−1?
YES
NO
状態値がIDのLUTを参照LUTとする
S4004
参照LUTを復号する
S4005
着目LUTと参照LUTを併合する
S4006
終了

フロントページの続き

(72)発明者　紺地　充
東京都大田区下丸子３丁目３０番２号　キヤノン株式会社内

審査官　大室　秀明

(56)参考文献　特開２００７－２２１７２０（ＪＰ，Ａ）
特開２００４－０２５５７２（ＪＰ，Ａ）
特開２００６－０８１２０９（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｈ０４Ｎ　　１／４６
Ｇ０６Ｔ　　１／００
Ｈ０４Ｎ　　１／６０