近接順位に基づくインデックスカラー画像の予測符号化

黒木修隆[†] 沼 昌宏[†] 山本啓輔[†]

本論文では新しい予測誤差の概念に基づくインデックスカラー画像の可逆符号化手法を提案する．一般にインデックスカラー画像に対して Lossless JPEG や CALIC を適用すると，(1) 予測関数が適切な値を算出できない，(2) 予測誤差がカラーパレット上の色の登録順序によって変化する，などの点から，高い圧縮率を得ることは困難であった．これらの問題点を解決するために本論文では，予測誤差の表現方法として近接順位を用いる符号化手法を提案する．近接順位とは，3 次元の色空間上に浮かぶ 1 つの色から見た他の色を，近さに基づいて順位付けすることにより得られる値である．提案手法では予測値と出現値の単純な差分を求めるのではなく，予測値に対する出現値の近接順位を求め，それをエントロピー符号化器に入力する．本手法により，カラーパレット上の色の登録順序には影響されない，高い圧縮率の符号化が可能になる．実験では，提案手法，BMP，GIF，PNG，Lossless JPEG，CALIC を用いてホームページ上のロゴやディザ処理された自然画像などを符号化し，提案手法の有効性を確かめた．

Predictive Coding of Palletized Images Based on Nearness Ordering

Nobutaka Kuroki,[†] Masahiro Numa[†] and Keisuke Yamamoto[†]

This paper presents a predictive coding technique for palletized images based on nearness ordering. Pixels in palletized images such as GIF formatted images with 256 colors are usually represented by index numbers in a color palette. To such images, it is impossible to get high efficient compression by using Lossless JPEG or CALIC because (1) the predictor does not work well for index numbers and (2) prediction errors depend on the order of colors in the palette. We propose a predictive coding with the idea of nearness ordering to overcome these problems. A nearness rank is a value assigned to each color based on the distance from the predicted color in 3-D color space. Our entropy coder encodes not prediction errors between predicted colors and original colors, but the nearness ranks assigned to these original colors. Experimental results of high compression ratios for palletized images, such as logos or dithering pictures, have shown the effectiveness of the proposed technique.

1. はじめに

ディジタル画像を画素あたりの bit 数（bit/pixel）によって分類すると，1，2，4，8，16，24 bit/pixel などに分けられる．これらのうち 16，24 bit/pixel の画像は，RGB の各成分の強度を数値として保持している場合が多く，それぞれハイカラー画像，フルカラー画像などと呼ばれている．一方，2，4，8 bit/pixel の画像は，カラーテーブルに基づいて，ピクセルの色をその管理番号（インデックス番号）で表現している場合が多く，インデックスカラー画像，color-mapped image，palletized image などと呼ばれている．インデックスカラー画像はフルカラー画像に比べてビデオメモリ上のデータ量が小さいため，WWW（World Wide Web）上で頻繁に用いられている．

インデックスカラー画像に対応した保存形式として，BMP[1]，GIF[2]，PNG[3]などがある．いずれの形式を用いても圧縮符号化が可能である．BMP ではランレングス符号化を行う圧縮モードと非圧縮モードのいずれかを選択できる．GIF と PNG では，それぞれ LZW[4]および LZ77[5]に基づく辞書ベース圧縮が行われている．

GIF よりも圧縮率の高い保存方法として，Palette ordering と予測符号化を組み合わせた手法が提案されている[6),7)]．この Palette ordering とは，予測符号化の効率を高めるために，カラーテーブル上で色の登録順序を変更する手法である．Handenfeldt ら[6)]は，予測誤差のエントロピーが最小となるような色の登録順序をシミュレーテッド・アニーリングにより探索する手法を提案した．また Memon ら[7)]は，処理時間短縮のために Pairwise Merge Heuristic と呼ばれる Palette

† 神戸大学工学部
Faculty of Engineering, Kobe University

ordering のアルゴリズムを提案し，これを Lossless JPEG[8]と組み合わせることで，GIF を上回る高い圧縮率を達成した．

しかし，これら Palette ordering ではインデックス番号の変更のみによって圧縮率向上を目指しており，各色を 1 次元のカラーテーブル上に並べるという点では従来手法と共通している．1 次元のインデックス番号は，色が持つ本来の輝度や色相を反映していない．よって，その値に対して予測符号化を適用しても，正しい予測が行われない可能性がある．

本論文ではインデックスカラー画像の予測符号化に 2 つの問題点があることを指摘し，これを解決する新しい符号化手法を提案する．提案手法は 3 次元の色空間において予測誤差を表現するために，近接順位という尺度を導入している．2 章では近接順位の定義，および近接順位に基づく予測符号化の実現方法を述べる．3 章では提案手法を従来の各手法と比較し，その圧縮能力を評価する．最後に 4 章でまとめを行う．

2. 近接順位を用いた予測符号化

2.1 インデックスカラー画像

8 bit/pixel のインデックスカラー画像は，一般に**表 1** に示すような色の表（カラーテーブル）に基づき，各画素の色をインデックス番号に置き換えて保持している．ここで色の登録順序はアプリケーションに依存するので，R，G，B の色成分の強度とインデックス番号の間に相関関係があるとは限らない．このような画像に対して，連続階調の画像（continuous-tone image）を対象にした予測符号化を行うと，次の 2 つの原因により十分な圧縮率が得られない．

(1) インデックス番号を利用した演算により，不適切な予測値を算出する．
(2) 予測値と出現したインデックス番号の差が，必ずしも視覚的な色の距離を反映しないために，予測誤差信号の電力が増加する．

この現象を Lossless JPEG[8]を用いた例で説明する．カラーテーブル上でインデックス番号が i である色を C_i（$i = 0, \cdots, 255$）とする．C_i は R，G，B の各信号の強度からなる 3 次元のベクトルであり，$C_i = (r_i, g_i, b_i)$ とする．表 1 のカラーテーブルに登録された色の，RGB 空間における配置を**図 1** に示す．たとえば予測関数として**図 2** の式 7) が用いられるとする．ここで画素 a，b の色が C_1 と C_3 ならば，これらのインデックス番号 $a = 1$，$b = 3$ を式 7) に代入すると $X = 2$ となるため，予測される色は C_2 となる．しかし，この予測関数によって本来求めたい色は C_1 と C_3 の中点，すなわち $C_0 = (0, 20, 10)$ である．このような不適切な予測の問題が (1) に対応する．

表 1 カラーテーブルの例
Table 1 Example of color table.

インデックス番号	R	G	B
0	0	20	10
1	0	20	0
2	0	30	10
3	0	20	20
～			
255	0	0	0

次に C_1 を予測値とし，C_{255} を出現値と仮定する．両者の RGB 空間におけるユークリッド距離はわずか 20 であるにもかかわらず，インデックス番号の差は 254 という大きな値になる．このようにインデックス番号間の差が，視覚的な色の距離を反映していないことが (2) の問題点である．

(1) の問題点は，インデックス番号を本来の色に変換してから計算することで解決できるが，(2) の問題点を解決するためには，予測誤差について新しい表現方法を導入する必要がある．そこで本論文では，視覚的に近い色どうしの差分をなるべく小さな値で表すために，近接順位という尺度を導入する．この近接順位を用いた予測符号化によって (2) の問題点を解決する．

2.2 2 色間の近接順位の定義

人間の視覚特性は複雑であり，3 次元の色空間に浮

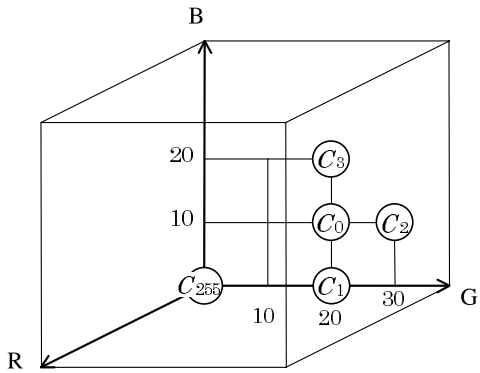


図 1 RGB 空間における色の配置
Fig. 1 Color locations in RGB space.

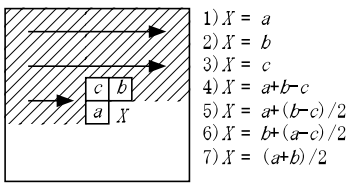


図 2 Lossless JPEG における予測関数
Fig. 2 Prediction formulae in lossless JPEG.

かぶ 2 つの色について，視覚的な距離を定義することは難しい．しかし，8 bit/pixel のインデックスカラー画像において使用されている色の数はたかだか 256 色にすぎず，ある点 C_i から見たすべての色 C_j（$j = 0, \cdots, 255$）に対して，近さの順位 0〜255 を割り当てることは容易である．そこで本節では，近接順位 $R(i,j)$ を定義する．

まず距離 $D(i,j)$ を C_i と C_j とのマンハッタン距離

$$D(i,j) = |r_i - r_j| + |g_i - g_j| + |b_i - b_j| \quad (1)$$

と定義する．これをユークリッド距離のように定義することも可能であるが，ここでは正確な距離を算出することが目的ではないので，計算式を簡略化した．距離 $D(i,j)$ は，0 から 765 までの整数値をとりうる．次に，ある C_i を基準色として，すべての C_j（$j = 0, \cdots, 255$）に対して，距離 $D(i,j)$ の小さな色から優先的に，0 から 255 までの順位を与え，それを近接順位 $R(i,j)$ と定義する．このとき，

$$R(i,j_1) < R(i,j_2) \Rightarrow D(i,j_1) \leq D(i,j_2) \quad (2)$$

が成立する．たとえば表 1 のカラーテーブルについて，C_2 を基準色として優先順位を付けた例を**図 3** に示す．自分自身には 0 を割り当て，$R(2,2) = 0$ とする．次に，C_2 から見て C_0 が最も近くに位置するので，$R(2,0) = 1$ を割り当てる．C_1 と C_3 のように等距離に位置する色が複数存在する場合は，インデックス番号の小さい方を優先し，$R(2,1) = 2$，$R(2,3) = 3$ とする．この手順を繰り返し，すべての C_j（$j = 0, \cdots, 255$）に対して，0 から 255 までの重複しない整数値を $R(2,j)$ として割り当てる．

このようにして得られた近接順位 $R(i,j)$ は，カラーテーブル上の色の登録順序にはほとんど影響を受けない値である．また，C_i と C_j から一意に $R(i,j)$ が求まり，逆に C_i と $R(i,j)$ から一意に C_j が求まるという性質がある．これらの性質を利用して予測符号化および復号を実現する．

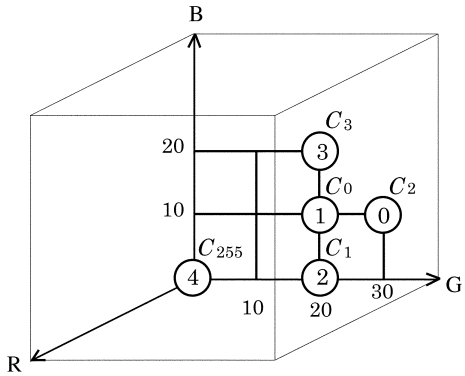


図 3 近接順位の割り当て
Fig. 3 Assignment of nearness rank.

2.3 近接順位行列 R の生成

後に述べる予測符号化では，予測値 C_i と出現値 C_j から予測誤差として $R(i,j)$ を出力する．逆に復号の際は，予測値 C_i と予測誤差 $R(i,j)$ から出現値 C_j を復号する．この処理は頻繁に行われるため，あらかじめ $R(i,j)$ を行列 R として準備しておくことにより，高速な符号化が可能になる．

2.2 節で述べた $R(i,j)$ の算出アルゴリズムを C 言語によって記述すると**図 4** のようになる．$D(i,j)$ および $R(i,j)$ は，それぞれ 2 次元配列 D[i][j] および R[i][j] と表されている．また，色 C_i の R，G，B 成分は，それぞれ C[i].r，C[i].g，C[i].b と表されている．本アルゴリズムでは，2 次元配列の各行に対して 1)〜4) からなる 4 パスの処理を行う．1) では式 (1) に従って D[i][j] の要素を計算する．たとえば表 1 のカラーテーブルに対する $D(i,j)$ は，**図 5** のようになる．2) から 4) では，バケットソートを応用することによって近接順位を求めている．2) で D[i][j] の分布を配列 histogram に格納し，3) でそれを累積することによって，距離 d に対応する順位を histogram[d] で表現する．最後の 4) で D[i][j] に対応する近接順位を histogram[D[i][j]] より求めるが，この際に同じ近接順位が複数の R[i][j] に割り当てられないように，R[i][j] に値を格納する直前に必ず histogram[D[i][j]] の値をデクリメントする．これによって 0 以上 255 以下の重複しない整数値が R[i][j] に割り当てられる．たとえば，図 5 の $D(i,j)$ に対する $R(i,j)$ は**図 6** のようになる．本アルゴリズムはカラーテーブル内の登録色数 n に対して，オーダ $O(n^2)$ の計算量を必要とする．

近接順位行列 R について注目すべき点は，それが画像の絵柄に関係なく，カラーテーブルの構造のみによって一意に決まる点である．したがって符号化器から復号器へカラーテーブル情報を伝送すれば，両者は

```
for( i = 0; i < 256; i++ ){

    for( j = 0; j < 256; j++ )  /* 1) */
        D[i][j] = abs( C[i].r - C[j].r )
                + abs( C[i].g - C[j].g )
                + abs( C[i].b - C[j].b );

    for( j = 0; j < 256; j++ )  /* 2) */
        histogram[ D[i][j] ]++;

    for( d = 1; d <= 765; d++ )  /* 3) */
        histogram[ d ] += histogram[ d-1 ];

    for( j = 255; j >= 0; j— )  /* 4) */
        R[i][j] = --histogram[ D[i][j] ];
}
```

図 4 C 言語による $R(i,j)$ 生成の例
Fig. 4 Example of making $R(i,j)$ by C.

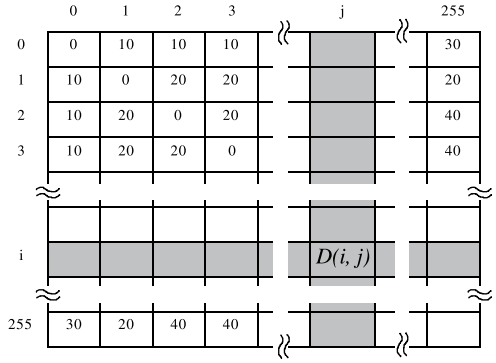


図 5 カラーテーブルから得られる距離 $D(i,j)$
Fig. 5 Distance $D(i,j)$ derived from color table.

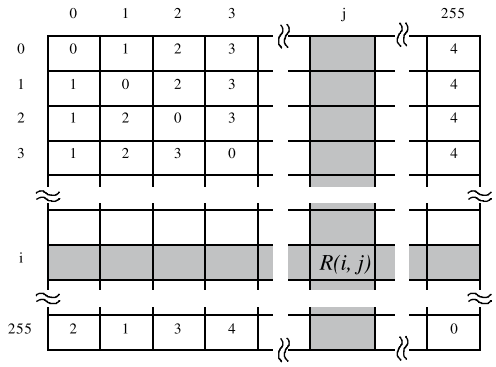


図 6 距離 $D(i,j)$ から導かれた近接順位 $R(i,j)$
Fig. 6 Nearness rank $R(i,j)$ derived from distance $D(i,j)$.

内部でまったく同じ R を算出できるため，R をオーバヘッドとして送る必要がない．また，近年のオペレーティングシステムで標準とされているカラーテーブルについては，事前に R を算出しておくことにより，より高速な符号化が可能になる．

2.4 符号化および復号のアルゴリズム

次に近接順位 $R(i,j)$ を利用した予測符号化のアルゴリズムについて述べる．提案手法の処理の流れを**図 7** に示す．本手法が Lossless JPEG と異なる点は，変換部において単純な差分を計算するのではなく，近接順位を算出する点のみである．予測部では最も簡単な前値予測，すなわち JPEG の予測関数 1) にあたる予測を行う．予測手法としてはこのほかにも，直上画素を用いる方法，出現頻度の高い色を選択する方法などが考えられるが，本論文では議論の対象を近接順位の導入効果に絞り，予測関数の最適化については今後の検討課題とする．変換部では近接順位を用いるが，これは復号側でもカラーテーブル情報から再構築できるため，オーバヘッドとして送出しない．符号化部で

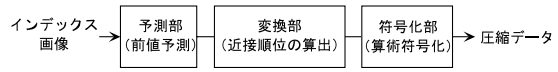


図 7 処理の流れ
Fig. 7 Coding process.

は JPEG で採用されている算術符号化と同じ処理を行う．この算術符号化は 1 パスで適応的に確率テーブルを更新しながら符号を生成するため，符号化テーブルなどのオーバヘッドを送出しない．符号化器の内部で用いられているパラメータについては，$U = 1$ および $L = 0$ とする．U および L は近傍画素の状態を 25 種類に分類して確率テーブルを切り換えるために用いられるが，詳細については文献 8) に譲る．

上の処理の具体的な手順を示す．インデックス画像について，画素 (x,y) のインデックス番号を $f(x,y)$ とする．

Step 1：最も左の画素 $(0,y)$ についてはインデックス番号 $f(0,y)$ をそのまま 8 bit で保存する．
Step 2：右の画素 (x,y) に移動し，そのインデックス番号を予測する．最も簡単な前値予測 $i = f(x-1,y)$ を用いる．
Step 3：予測値 i と出現値 $j = f(x,y)$ より，近接順位 $e = R(i,j)$ を得る．
Step 4：e をエントロピー符号化器に入力する．
Step 5：1 ラインの処理が終了するまで Step 2〜4 を繰り返す．
Step 6：すべてのラインについて Step 1〜5 を繰り返す．

上の符号化手順に対応する復号の手順を以下に示す．

Step 1：8 bit の値を読み込み，最も左の画素 $(0,y)$ のインデックス番号 $f(0,y)$ を復号する．
Step 2：右の画素 (x,y) に移動し，そのインデックス番号を予測する．符号化時と同じ予測関数によって $i = f(x-1,y)$ を得る．
Step 3：エントロピー復号器より近接順位 e を受け取る．
Step 4：e に等しい値を 2 次元配列 $R(i,j)$ の中から探し出し，列番号 j を注目画素 (x,y) のインデックス番号 $f(x,y) = j$ とする．
Step 5：1 ラインの処理が終了するまで Step 2〜4 を繰り返す．
Step 6：すべてのラインについて Step 1〜5 を繰り返す．

以上のアルゴリズムは $R(i,j)$ が既知である場合はいっさいの四則演算を行わないので，非常に簡単かつ高速である．このような簡単な手法で高い圧縮率が得

表 2 減色された標準画像
Table 2 Standard images with 8 bit colors.

画像名	減色方法	Palette ordering	サイズ（画素）	色深度（ビット/画素）	使用色数
Airplane1	最近隣色量子化	×	256 × 256	8	254
Lena1	〃	×	〃	8	255
Mandrill1	〃	×	〃	8	254
Peppers1	〃	×	〃	8	255
Airplane2	最近隣色量子化	○	〃	8	254
Lena2	〃	○	〃	8	255
Mandrill2	〃	○	〃	8	254
Peppers2	〃	○	〃	8	255
Airplane3	誤差拡散法	×	〃	8	254
Lena3	〃	×	〃	8	255
Mandrill3	〃	×	〃	8	254
Peppers3	〃	×	〃	8	256
Airplane4	誤差拡散法	○	〃	8	254
Lena4	〃	○	〃	8	255
Mandrill4	〃	○	〃	8	254
Peppers4	〃	○	〃	8	256

られることを，次章で示す．

3. 実　　験

提案手法の有効性を確かめるため，

- フルカラーから 256 色以下に減色された標準画像
- 全国の大学のホームページから収集した 100 枚の GIF 画像

を準備し，それらを様々な形式で保存する．そしてファイルサイズ，圧縮率，処理時間，およびメモリ使用量を評価する．実験はすべて IBM-PC 互換機（Pentium II 300 MHz）の上で行った．

3.1 減色された標準画像に対する圧縮率

フルカラーの標準画像 Airplane，Lena，Mandrill，Peppers を画像加工ツール Photoshop 4.0 によって 256 色以下に減色する．このとき，最近隣色量子化，誤差拡散法[9]，Palette ordering の使用/不使用など様々な条件を与え，**表 2** に示す 16 枚の画像を作成した．Palette ordering の方法としては Memon [7] らが提案した Pairwise Merge Heuristic を用いた．これらの画像を BMP [1]，GIF [2]，PNG [3]，Lossless JPEG [8]，CALIC [10]，および提案手法を用いて保存し，ファイルサイズおよび圧縮率を測定する．Lossless JPEG においては図 2 の予測関数 1) および 7) を用いた保存形式を，それぞれ JPEG 1) および JPEG 7) と表す．

まず，最近隣色量子化された画像 Airplane1〜Peppers1 のファイルサイズおよび圧縮率を**表 3** に示す．圧縮率 K は，平均ファイルサイズを S（byte），画素数を N とすると，

$$K = (1 - S/N) \times 100\% \qquad (3)$$

のように計算した．すなわち K が大きいほど圧縮効果が高いことを意味する．すべてのファイルは 256 色のカラーテーブル情報を含んでいる．BMP については非圧縮なので，カラーテーブル情報がオーバヘッドとなり，圧縮率が負の値となる．提案手法は PNG を 6 ポイント上回り，44.5%の高い圧縮率を示した．その他の手法は 30%程度の圧縮率にとどまった．JPEG 1) よりも JPEG 7) の方が低い圧縮率を示す現象は文献 7) でも報告されており，本論文で指摘した (1) の問題点が表れていると考えられる．

次に最近隣色量子化に加えて Palette ordering を行った画像 Airplane2〜Peppers2 のファイルサイズおよび圧縮率を**表 4** に示す．PNG および GIF は Palette ordering を行う前と後でまったく圧縮率が変化しないが，JPEG 1)，JPEG 7)，および CALIC は 6 ポイント以上圧縮率が改善され，GIF を上回る結果となった．ただし JPEG 1) よりも JPEG 7) の方が低い圧縮率を示す現象は改善されていない．すなわち (1) の問題点は PaletteOrdering によって解決できる性質のものではない．一方，提案手法は Palette ordering の使用/不使用にかかわらず，44%前後の高い圧縮率を得ている．提案手法はパレットの登録順序にほとんど影響を受けない近接順位行列に基づいて符号化を行うため，安定した圧縮率を得ることができる．

次に誤差拡散法が施された画像 Airplane3〜Peppers3 のファイルサイズおよび圧縮率を**表 5** に示す．誤差拡散法は視覚的に良好な画像を生成するが，画素値に微少な振動を与えるため，圧縮が困難になることは容易に予想できる．実験の結果，BMP 以外のすべての手法について表 3 よりも圧縮率が 6.4 ポイント以上低下した．特に予測符号化を基本とする JPEG 1),

表 3 最近隣色量子化された標準画像のファイルサイズ
Table 3 File sizes for standard images after Nearest-Neighbor mapping.

手法	BMP	GIF	PNG	JPEG 1)	JPEG 7)	CALIC	提案手法
Airplane1	66,614	34,551	32,293	38,489	43,713	41,239	29,706
Lena1	66,614	45,957	40,201	47,974	51,752	49,851	37,380
Mandrill1	66,614	58,567	52,868	55,154	56,923	55,211	47,296
Peppers1	66,614	40,388	35,945	39,822	44,008	40,747	31,059
平均ファイルサイズ	66,614	44,866	40,327	45,360	49,099	46,762	36,360
圧縮率 K	−1.6%	31.5%	38.5%	30.8%	25.1%	28.6%	44.5%

表 4 最近隣色量子化 + Palette ordering された標準画像のファイルサイズ
Table 4 File sizes for standard images after Nearest-Neighbor mapping + Palette ordering.

手法	BMP	GIF	PNG	JPEG 1)	JPEG 7)	CALIC	提案手法
Airplane2	66,614	34,551	32,281	30,752	31,954	30,495	30,204
Lena2	66,614	45,957	40,193	43,894	44,598	42,399	37,828
Mandrill2	66,614	58,567	52,856	53,546	54,624	51,495	47,590
Peppers2	66,614	40,388	35,943	37,198	39,597	38,135	31,463
平均ファイルサイズ	66,614	44,866	40,318	41,348	42,693	40,631	36,771
圧縮率 K	−1.6%	31.5%	38.5%	36.9%	34.9%	38.0%	43.9%

表 5 誤差拡散処理された標準画像のファイルサイズ
Table 5 File sizes for standard images after Error-Diffusion.

手法	BMP	GIF	PNG	JPEG 1)	JPEG 7)	CALIC	提案手法
Airplane3	66,614	38,774	36,618	47,333	52,524	49,827	34,988
Lena3	66,614	51,258	45,945	55,582	59,665	60,091	43,045
Mandrill3	66,614	60,565	54,357	57,407	58,811	57,295	48,867
Peppers3	66,614	45,855	41,810	47,905	51,764	48,779	36,752
平均ファイルサイズ	66,614	49,113	44,683	52,057	55,691	53,998	40,913
圧縮率 K	−1.6%	25.1%	31.8%	20.6%	15.0%	17.6%	37.6%

表 6 誤差拡散処理 + Palette ordering された標準画像のファイルサイズ
Table 6 File sizes for standard images after Error-Diffusion + Palette ordering.

手法	BMP	GIF	PNG	JPEG 1)	JPEG 7)	CALIC	提案手法
Airplane4	66,614	38,774	36,599	36,205	37,397	34,855	35,434
Lena4	66,614	51,258	45,934	49,909	50,610	48,019	43,441
Mandrill4	66,614	60,565	54,347	55,685	56,629	53,679	49,157
Peppers4	66,614	45,855	41,797	44,305	46,471	43,771	37,181
平均ファイルサイズ	66,614	49,113	44,669	46,526	47,777	45,081	41,303
圧縮率 K	−1.6%	25.1%	31.8%	29.0%	27.1%	31.2%	37.0%

JPEG 7)，および CALIC は圧縮率が 10 ポイント以上低下しており，誤差拡散法の影響を受けやすい．この原因は，予測関数の的中率が低下すると (2) の問題点が顕著に現れるためである．一方，提案手法において予測関数が JPEG 1) と同じであるにもかかわらず圧縮率の低下が小さい理由は，予測後の処理，すなわち近接順位に基づく符号化が適切であるためと考えられる．提案手法ではたとえ予測値が外れても近接順位が極端に大きな値にならないため，圧縮率の低下を抑えることができる．

次に誤差拡散法に加えて Palette ordering を行った画像 Airplane4〜Peppers4 のファイルサイズおよび圧縮率を**表 6** に示す．PNG および GIF は先程と同様に Palette ordering を行う前後でまったく圧縮率が変化しない．JPEG 1)，JPEG 7)，および CALIC は 8.4 ポイント以上圧縮率が改善されているが，PNG の圧縮率 31.8%にはわずかに至らない．それに対して提案手法は Palette ordering の使用/不使用にかかわらず，37%以上の高い圧縮率を得ている．以上のことから，提案手法は様々な条件の減色画像に対して有効であることが分かる．

3.2 100 枚の GIF 画像に対する圧縮率

全国の大学のホームページより収集した 100 枚の GIF 画像を提案手法によって再び圧縮保存し，GIF に対する圧縮率 K_g を求める．圧縮率 K_g は，元の GIF 形式のファイルサイズを S_g，提案手法によるファイルサイズを S とすると，

$$K_g = (1 - S/S_g) \times 100\% \quad (4)$$

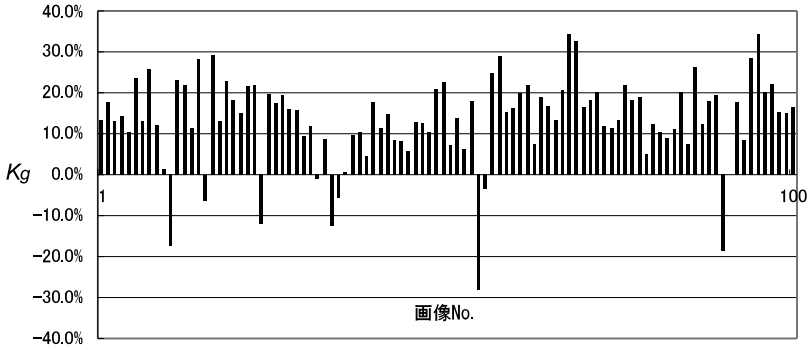


図 8 100 枚の GIF 画像に対する圧縮率
Fig. 8 Compression ratios for 100 GIF images.

のように計算する．実験結果を**図 8** に示す．横軸は 100 枚の画像を表し，縦軸は圧縮率 K_g を表す．K_g が負の場合は提案手法によるファイルサイズの方が大きくなり，圧縮できなかったことを意味する．実験では 100 枚中 9 枚の画像を圧縮できなかったが，残りの 91 枚の画像を圧縮し，最大で 34.3%，平均で 13.5%の圧縮率を得ることができた．提案手法はすべての画像について GIF を上回るとは限らないが，現在インターネット上で流通している多くのインデックス画像に対して高い圧縮率が期待できる．

3.3 処 理 時 間

最も予測関数が簡単な JPEG 1) と提案手法について，画像 Lena を符号化する際の処理時間を**表 7** に示す．表より，予測部および変換部については両者とも非常に高速であることが分かる．提案手法は 2 次元配列 R[i][j] に一度アクセスするのみで近接順位を求めるため，単純に差分を計算する JPEG 1) と比べても，処理時間がほとんど変わらない．むしろ，それ以外の処理時間が全体に大きく影響する．文献 7) で提案されているように JPEG 1) と Palette ordering を組み合わせると，処理時間が 67 s と膨大になり，非実用的であることが分かる．Palette ordering を省略すれば処理時間を 0.31 s 以内に抑えられるが，その場合は事前に何らかのアプリケーションによって適切なカラーテーブルを構成しておく必要がある．

一方，提案手法においては全体の処理時間 1.08 s のうち，0.78 s が近接順位行列 R の生成に費やされている．ただし 2.2 節で述べたように，代表的なカラーテーブルについては R を事前に求めておくことができるので，そのような場合は処理時間を 0.31 s 以内に抑えることができる．以上の処理時間から提案手法は十分に実用的な手法と考えることができる．

表 7 処理時間
Table 7 Computation times.

手順	JPEG 1)	提案手法
パレット・オーダリング	67 s	—
近接順位行列の生成	—	0.78 s
予測部・変換部の処理	0.0055 s	0.0061 s
符号化部の処理	0.30 s	0.29 s
合計	67 s	1.08 s

3.4 実行時のメモリ使用量

最後に本実験で用いた各種のソフトウェアが計算機上で必要とするメモリ量を**表 8** に示す．処理する画像のサイズは 256×256 画素とした．表の値はコンパイラの性能およびプログラムの記述方法に依存するため，これによって各保存形式の優劣を決めることは適切ではないが，いずれのソフトウェアも 1〜3 MByte 程度のメモリを必要としており，極端な差はないことが分かる．提案手法は Lossless JPEG に次いで少ないメモリ使用量であり，両者の差はたかだか 148 Kbyte である．この差の理由は，提案手法が Lossless JPEG に比べて近接順位行列（約 128 Kbyte）を確保するために余分なメモリを必要とするからである．しかし，この増加分が全体に占める割合は約 11%である．以上のことから，提案手法は十分に実用的であるといえる．

4. ま と め

本論文では新しい予測誤差の概念である近接順位に基づくインデックスカラー画像の可逆符号化手法を提案した．提案手法を減色された標準画像やホームページ上の画像などに適用し，その高い圧縮率を確かめた．実験の結果，GIF に比べて約 12 ポイント，Lossless JPEG に比べて約 7 ポイント，PNG や CALIC に比べて約 5 ポイントの圧縮率を改善することができた．また誤差拡散法や Palette ordering が施された様々な

表 8 実行時のメモリ使用量
Table 8 Memory usage.

実行形式ファイル名	用途	動作 OS	メモリ使用量（Kbyte）
PaletteOrdering.exe	Palette ordering の処理	Windows98	1,636
gif2png.exe	GIF から PNG への変換	Windows98	2,572
JPEG.exe	Lossless JPEG による保存	Windows98	1,048
encode16.exe	CALIC による保存	Windows98	3,232
IndexCompress.exe	提案手法による保存	Windows98	1,196

画像に対しても有効であることが確認できた．

本論文では議論を近接順位の導入効果に絞り，予測関数の比較評価を行わなかった．しかし，この予測関数の最適化によって，さらなる圧縮率の向上が期待できる．今後は出現頻度の高いインデックス番号を予測値とする方法や，近傍画素の状態によって予測関数を選択する手法などを導入し，提案手法の効果を確認することが課題である．

参 考 文 献

1) Microsoft Corp.: *Microsoft Windows Programmer's Reference*, Microsoft Press (1990).
2) CompuServe Inc.: *Graphics Interchange Format (GIF) Specification* (1990).
3) Atzberger, P. and Zolli, A.: Portable network graphics, *WEB Tech.*, Vol.1, pp.65–68 (1996).
4) Welch, T.: A Technique for High-Performance Data Compression, *IEEE Computer*, Vol.17, pp.8–19 (1984).
5) Ziv, J. and Lempel, A.: A Universal Algorithm for Sequential Data Compression, *IEEE Trans. Inf. Theory*, Vol.23, pp.337–343 (1977).
6) Hadenfeldt, A.C. and Sayood, K.: Compression of Color-Mapped Images, *IEEE Trans. Geosciences and Remote Sensing*, Vol.32, pp.534–541 (1994).
7) Memon, N.D. and Venkateswaran, A.: On ordering color maps for lossless predictive coding, *IEEE Trans. Image Processing*, Vol.5, pp.1522–1527 (1996).
8) International Organization for Standards /International Electrotechnical Commission (ISO/IEC): *Digital Compression and Coding of Continuous-tone Still Image*, International Standard 10918-1 (1992).
9) Floyd, R. and Steinberg, L.: An Adaptive Algorithm for Spatial Gray Scale, *SID International Symposium Digest of Technical Papers*, Vol.4.3, pp.36–37 (1975).
10) Wu, X. and Memon, N.: CALIC–A Context Based Adaptive Lossless Image Codec, *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference*, pp.1890–1893 (1996).

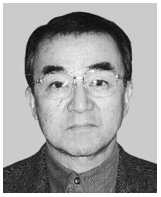
(平成 13 年 10 月 16 日受付)
(平成 14 年 9 月 5 日採録)

黒木 修隆

1990 年神戸大学工学部電子工学科卒業．1995 年同大学大学院自然科学研究科博士課程修了．同年同大学工学部電気電子工学科助手．工学博士．画像処理，LSI 設計に関する研究に従事．電気学会，電子情報通信学会各会員．

沼 昌宏（正会員）

1983 年東京大学工学部精密機械工学科卒業．1985 年同大学大学院修士課程修了．同大学助手を経て 1989 年同大学講師．1990 年神戸大学大学院自然科学研究科講師．1995 年同大学工学部電気電子工学科助教授．工学博士．主に LSI CAD，FPGA 応用，画像処理に関する研究に従事．IEEE，ACM，電子情報通信学会各会員．



山本 啓輔

1962 年神戸大学工学部電気工学科卒業．同年松下電器産業（株）入社．主としてテレビ受信機の開発，研究に従事．2000 年 5 月より神戸大学工学部電気電子工学科教授．工学博士．放送と通信の融合，画像処理，LSI CAD に関する研究に従事．映像情報メディア学会会員．