多値画像のＤＦ符号化について

7J-7

鎌田　清一郎　　江川　　護　　河口　英二
九州工業大学　工学部

1．はじめに

ビットプレーン分解による情報圧縮手法は古くから様々の形で試みられている．その中で，ビットプレーン上で重要な部分を保存し，重要でない部分を捨てる方法を考察し，その一手法として，ＤＦ表現を利用した情報圧縮法を提案してきた[1]．本稿ではその後処理について，すなわち，ＤＦ表現によって得られた記号"0","1","(" に対しての符号化問題においてハフマン符号化を用いた方法について報告する．

2．ビットプレーン分解とＤＦ表現を利用した画像の情報圧縮

2.1 ＤＦ表現

二値画像のＤＦ表現[2]を以下に説明する．$2^R \times 2^R$の二値画像を４分木表現すると，図１のようになる．ＤＦ表現は４分木の終端ノードを"0"または"1"で置き換え，また，非終端ノードを"(" で表記する方法である．記号化の順番は左下，右下，左上，右上である．図１のＤＦ表現は((0111(1(101010(11(110000となる．また，４分木に現れる部分画面をΓ^r_αとする．ここで，ｒは部分画面の大きさを表す"画面の次数"，αは"部分画面のアドレス" である．r=0は全画面を表す．例えば，ＤＦ表現上の最も左と最も右にある0に相当する部分画面はΓ^2_{0000}とΓ^1_{11} である．さらに，複雑さの尺度は

$$Cp = \frac{\text{終端ノードの個数}}{4^R} \qquad (1)$$

で表す．

2.2 画像情報圧縮の概要

多値画像$2^R \times 2^R$をビットプレーン分解するとN枚のビットプレーンが生成されるとする．これまで，このビットプレーン上で如何なる情報を取捨選択するかを考察してきた．ここではその概要を示す．

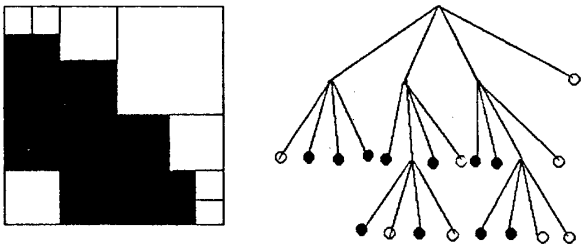
多値画像のビットプレーン$\Gamma(1),\Gamma(2),\cdots,\Gamma(N)$に対して，グレイコードによるビットプレーンを次式により導出する．

$$\Gamma'(1) = \Gamma(1)$$

$$\Gamma'(n+1) = \Gamma(n) \oplus \Gamma(n+1), \quad n=1,2,\cdots,N-1$$

ただし，$\oplus$は排他的論理和である．

画像データの中で，人間にとって有意な情報はエッジ上の不連続点のような特徴点に集中していると考えられる．ビットプレーンを上位から下位へと順に見ていくと白黒境界点の密集している部分が次第に多くなっていく．図２に下位ビットプレーンと不連続点の分布を示す．下位ビットに見られる不連続点の密集している部分は，それぞれのビットプレーンでは"ノイズ"のように見え，あまり重要ない情報と思われる．従って，"或る領域内で不連続点が或る一定以上に密集した部分の情報は捨て，不連続点の疎な部分はもとのまま保存する" ことは人間の一視覚特性に合った情報圧縮法であると考えられる．そこで，取捨選択の指標としては，n番目のビットプレーンの部分画面に対して



(a) 二値画像　　(b) ４分木表現

図１　二値画像と４分木表現

(a) 下位ビットプレーン　　(b) 不連続点の分布

図２　下位ビットプレーンと不連続点の分布

The DF-Coding for Multi-valued Images

Seiichiro KAMATA, Mamoru EGAWA and Eiji KAWAGUCHI
Kyushu Institute of Technology, Faculty of Engineering

$$P = C_p(\Gamma_{\alpha}'(n)) - (C_p(\Gamma_{\alpha}(n))-C_p(\Gamma_{\alpha}(n-1))>C_{TH}$$

を利用する．ただし，C_{TH}は予め定めた閾値である．

この具体的な情報の捨て方は上述の"ノイズ"のように見える領域を単一色（灰色）の領域に置き換える操作であり，これは原画像をその視覚的な特徴を保存しながら，しかも符号化効率の高い画像になるように原画像を部分的に変更することである．この操作を「画一化操作」と呼ぶ．

3．画一化操作後における符号化

これまでのＤＦ符号化は記号"0","1","("に対して

$$\begin{cases} \text{"("} \rightarrow 11, \\ \text{"0"} \rightarrow 0, \\ \text{"1"} \rightarrow 1, & r=R\text{のとき} \\ \qquad\ 10, & r\neq R\text{のとき} \end{cases}$$

と行っていたが，画像によってはＤＦ表現を行った場合にかえって圧縮効率が悪くなる場合がある．そこで，これまでのＤＦ表現に対して，記号"[" を加える．これはある部分画面のＤＦ表現データがビットプレーン上の原データよりも長くなるときに，原データをそのまま用いるための開始記号である．この場合，原データの個数は"("の場合と同様に"["の現れた位置の次数によって一意に定まる．例えば，ある部分画像が市松模様に近い場合，ＤＦ符号化よりも原データを用いる方がよい．これは次のような変換である．

$$\cdots\cdots\overbrace{((10\cdots(1001}^{4^r+(4^r-1)/3}\cdots\cdots$$

$$\downarrow$$

$$\cdots\cdots\overbrace{[10\cdots1010}^{4^r+1}\cdots\cdots$$

実際の符号化は記号の個数が$4m+1$(mは自然数)となる．このとき，２個ずつ記号の符号化を行うことによって符号化効率をあげる．この組合せの中で起こり得ない符号があり，16通り中，表１に示したように14通りである．符号化に際しては各記号ごとに分布をとり，ハフマン符号化を適用した．

4．実験

この実験においては圧縮効率をあげるため，以下のように行った．画像データはGIRLであり，一画素当り８ビット，512X512画素である．この画像データに対して符号化実験を行った．従来のＤＦ符号化と本ＤＦ符号化の比較結果を第１～４ビットプレーンに対して表２に示す．

表１　記号と符号の対応

記号	符号	記号	符号
00	101	01	001
0(	100	0[	000010
10	110	11	011
1(	1111	1[	1110010
(0	000	(1	0111
((	1010	([	0110010
[0	100010	[1	010010

表２　圧縮効率

ビット・プレーン番号	従来のＤＦ符号(ﾊﾞｲﾄ)	本ＤＦ符号化(ﾊﾞｲﾄ)
1	7471	6642
2	14849	13105
3	30244	24547
4	34856	25565
計	87421	69859

この結果，符号化としては約２割の圧縮効率を上げることができた．

ビットプレーンごとに，符号内に含まれる原データの割合は 0.30, 0.31, 0.62, 0.96 と段々と高くなっている．また，圧縮効率を見ると，0.89,0.88,0.81, 0.73となり，下位ビットプレーンに進むにしたがって良くなっている．これは従来のＤＦ符号化が下位ビットプレーンの符号化に適していないが，それが改善されていることが確認された．

5．まとめ

本稿では多値画像のビットプレーン分解からのＤＦ表現によって得られた記号"0","1","(" に対しての符号化問題においてハフマン符号化を用いた方法について述べた．

参考文献

(1) 鎌田，河口:"ﾋﾞｯﾄﾌﾟﾚｰﾝ分解による多値画像の情報圧縮に関する考察",信学論(DII),J72-DII,7,1989.

(2) 谷口，河口:"二値画像の複雑さと多値画像の閾値処理に関する考察",信学論(D),J70-D,1,1987.