ディザパターンを用いたスクリーン画像圧縮技術

Screen Image Compression Technology Using Dither Patterns

山　本　敏　嗣＊
Toshitsugu YAMAMOTO

要旨

プリンター開発では速度性能向上とコストダウンの両方が求められており，加えて高画質化のために多ビット化が進んでいる。このような要求とデータ処理とを共存させるためには高性能な圧縮技術が必要である。我々は，従来新しい発想が困難といわれるロスレス圧縮技術領域で，画像とディザパターンの関係に着目し，新規な圧縮技術CDI（compression algorithm for dithered image）を開発した。4bitスクリーン画像に対して1pixel/cycleの高速処理性能を実現しつつ，従来技術と比較して圧縮性能を平均20%向上，回路規模を約1/3にし，搭載ASIC（application-specific integrated circuit）のコストダウンを実現した。さらに，画像の種類による圧縮後サイズの変動も小さくなった。本技術は，今後発売されるプリンター及びプリンターベースの複合機に搭載される予定である。

Abstract

In the development of printers, both faster printing and lower costs are highly in demand, as is the higher image quality that an increasing number of bits can achieve. To satisfy these demands as they figure with data processing, image compression technologies exhibiting high performance are essential.

It has been assumed by many in the field of lossless compression technology that no further innovation could be expected in the field. But by focusing on the relationship between an image and a dither pattern, we have developed a new lossless compression technology called “CDI” (compression algorithm for dithered images).

With CDI, we achieved the high-speed processing property of one pixel/cycle per 4-bit screen image. We also improved compression performance by 20% on average, while shrinking the circuit size to about 1/3 and reducing the cost of the incorporated ASIC (application-specific integrated circuit). Further, after image compression and across various image types, we achieved image size variation that was minimal.

This technology will be incorporated into Konica Minolta printers and printer-based multi-function printers bound for the market.

＊コニカミノルタテクノロジーセンター㈱
システム技術研究所　イメージシステム開発室

1　はじめに

近年の厳しいプリンター開発競争の中，性能向上とコストダウンを実現する技術が要望されている。高画質を実現するためスクリーンデータの多ビット化が進み，データ量が増大している一方，さらなる多機能化に対応するため，画像データに割けるメモリが少なくなってきている。加えて，コストダウンのために安価なCPUの利用が求められ，処理速度の改善が困難になっていた。その解決には画像圧縮技術の導入があるが，他社技術を用いると，ロイヤリティの負担や生産上の制限などが発生し，求められるコストの中では設計が難しくなる課題があった。そこで，我々は独自の画像圧縮技術を開発することにした。

一般に，画像圧縮はデータのモデル化と符号化の2段階で行われる。高性能な圧縮技術の開発には特にモデル化の開発が肝要である[1)]。

これまでに開発された主な圧縮アルゴリズムは，コンテキスト圧縮と辞書式圧縮2つに集約される[2)]。

コンテキスト圧縮は圧縮対象となる画素値を周辺画素（コンテキスト）から予測し，予測結果との相違を求めてモデル化し，算術符号化によって符号化を行う。しかし，算術符号化がボトルネックになって高速化が難しく，またコンテキストと注目画素の事象の数が増大するため，多ビット化対応も困難であった。

辞書式圧縮は圧縮対象となるデータ列と同じパターンをすでに圧縮したデータ列から探し，その位置情報を求めてモデル化して符号化を行うことで圧縮する。多ビット化が容易で符号化方式によらず高い圧縮率が得られる。しかし，パターンマッチング処理がボトルネックになり，特に小規模ハードウェアで圧縮を実現するのが困難であった。

そこで我々は，圧縮対象であるスクリーンデータに最適で，かつ高速に符号化できるモデル化を考案し，全く新しい原理の圧縮技術を開発した。従来のロスレスの圧縮では必須であったデータの統計処理を廃し，少ない計算量で高い圧縮率が得られるようにした。この結果，圧縮機能にかかるコストを削減しながら，プリンターの高性能化を実現した。本稿ではその技術の概要を紹介する。

2　圧縮原理と特徴

ディザパターンを用いてハーフトーン処理を行った画像データは，データのほとんどがディザパターンに起因して，特定の配列をしている。

ディザパターンに一致している*ハーフトーン画像は，その部分の濃度と，ハーフトーン処理に使用したディザパターンを用いれば，完全に復元できる。この考えに基づく圧縮方法をCDI (Compression algorithm for dithered image)と呼ぶ。

CDIでは以下のステップで圧縮が行われる。

①ディザパターンを用いてハーフトーン化した画像をブロックに分割する（大ブロック）。
②その部分がディザパターンと一致しているかチェックする。一致していれば濃度に相当する代表値をそのパターンに与えハフマン符号化して出力する。
③ディザパターンと一致しないブロックは，以前に処理済みのデータに同じパターンがなかったか検索する。同じパターンが見つかったらその位置データを圧縮データとして出力する。①の処理で一致していればそちらを優先する。
④ディザパターンとも処理済みデータとも一致しなかった場合，大ブロックをさらに小さい小ブロックに分割しディザパターンと一致しているかチェックする。
⑤ディザパターンと一致している小ブロックについては濃度に相当する代表値をそのパターンに与えハフマン符号化して出力する。
⑥ディザパターンと一致しない小ブロックについては，1バイトごとに画像データ自体をハフマン符号化する。

CDIはブロック単位で圧縮し，ラインごとに展開するブロック圧縮ライン展開型の圧縮法になっている。今回，ハードウェアでの圧縮展開を考慮して，Fig.1 に示すようにブロックの縦サイズを1としている。こうすることで1次元の圧縮になり，圧縮展開ともラインメモリーなしで処理できる。

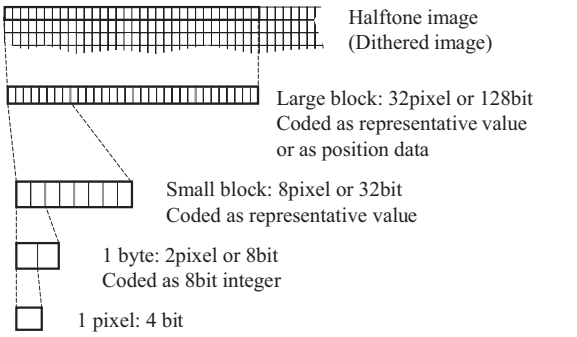

Fig.1 Large and small blocks to which compressed code is assigned in 4-bit dithered image

＊ここでは，ハーフトーン（処理済）画像がディザパターンに起因する配列をしているとき，「ディザパターンに一致している」と表現する。

2. 1　代表値算出と処理

2値化されたハーフトーン画像とは，各々の画素にドットを打つか打たないかを表現したものである。ディザハーフトーン画像は元の画像データを各々の画素に設けられた閾値（ディザパターン）と比較して，値が閾値以上のときドットを打ち，閾値未満のとき打たないと判断して得られる。

ある画素にドットを打つ場合，元の画像データは適用されたディザパターンの閾値以上であった，すなわち，この画素の下限値はディザパターン閾値であったことが判る。逆にドットを打たない場合は，元の画像データはディザパターンの閾値未満であった，すなわちこの画素の上限値はディザパターンの閾値－1であったことが判る。

一定の広さを持った領域の各々の画素について，上限値，下限値を集計し，上限値の最小値Jsと下限値の最大値Ksを求める．それらが，

$$Js \geqq Ks$$

であったとき，この範囲のドット配列はディザパターンどおりであったといえる。ディザパターンどおりにドットが配列しているのだから，

$$Js \geqq r \geqq Ks$$

であるようなr（一定）の値を持った画像データを元のディザパターンで処理すると，与えられたドットパターンを完全に再現することができる。したがって，その領域の画像データを再現するためには，ただひとつの代表値rだけを記録すればよい。Fig.2 にその仕組みを示した。

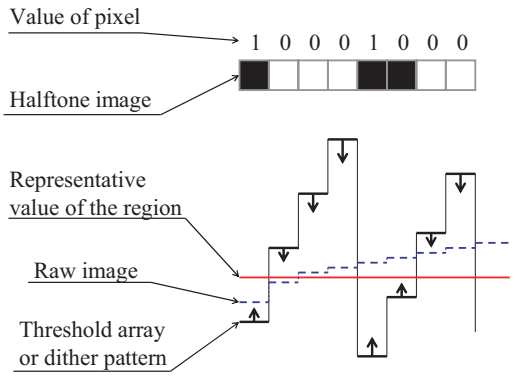

Fig.2 Representative value that generates 1-bit dithered image using threshold array

多値ハーフトーン画像の場合も，Fig.3 に示すように，2値ハーフトーン画像と同様に考えることができる。ハーフトーン画像のある画素値がhであったとすると，ディザパターンと比較して，ハーフトーン処理前の画像データの上限値下限値を決めることができる。

これを集計して

$$Js \geqq r \geqq Ks$$

であるような代表値rを使えば，このrのみでハーフトーン画像を再現することができる。

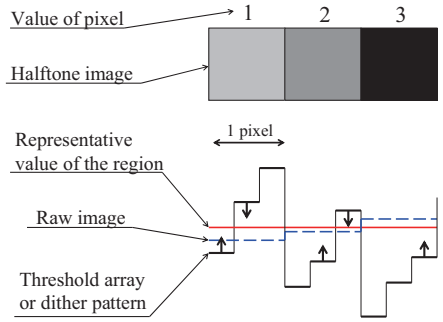

Fig.3 Representative value that generates 2-bit dithered image using threshold array

以上に述べた代表値算出ではパターンとの一致を判定するのに，パターンマッチングによる探索ではなく注目部分の大小比較だけで行うことができるので，高速化が容易になる。

この代表値rは，元画像の濃度データに関連しているデータであり，8ビットのデータ量を持っている。該当ブロックを128 ビットブロックとすると，128ビットを8ビットで表せるので，概ね1/10に圧縮できたことになる。

こうして求めた代表値rはその近隣の領域の代表値と関連している。従って，直前の領域の代表値との差分を取ると，頻度の偏りは大きくなると予想される。実際に差分の分布を調べてみたところ，Fig.4 に示すようになり，画像の種類によらず概ね同様になることが判った。

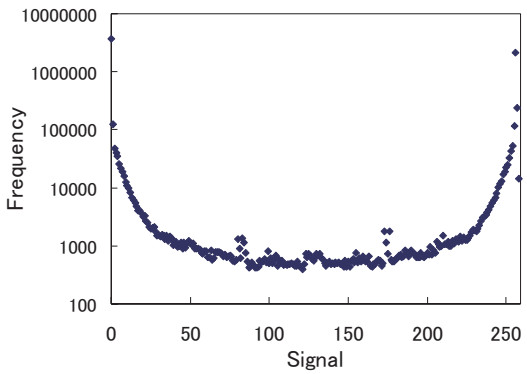

Fig.4 Distribution of signal frequency in CDI

偏りが非常に大きく，分布も概ね一定であるデータは，固定ハフマン符号化によって，高速に大きく圧縮することができる。画像にもよるが，概ね2 ～ 3ビット程度に圧縮される。

1ブロック128ビットであったデータがハフマン符号によって2 ～ 3ビットで表現できるので，データは数十分の1に圧縮されたことになる。

2. 2　繰り返しデータ配列の処理

アプリケーションソフトによっては透明度を設定できない描画系で半透明の描画を実現するために，半透明オブジェクトと背景のオブジェクトの画像データを交互に出力する場合がある。このようなアプリケーションで半透明画像を処理した場合，数画素程度の短い周期で画像が切り替わるため，この部分ではハーフトーン画像のデータとディザパターンとがほとんど一致せず，結果として圧縮率が低下する。

圧縮率低下を避けるため，ディザパターンと一致しない場合は，すでに圧縮した画像データ内に，現在のブロック内のデータと一致するものがないか検索する。一致するパターンが見つかった場合は，一致位置を圧縮データとする。

この処理は，ディザパターンとの一致が見られない場合の例外的な処理なので，辞書式圧縮と比較して，固定長の狭い範囲での探索で十分な効果をあげることができる。結果として，この場合の圧縮率は1/25 ～ 1/10程度となった。

2. 3　ディザパターンとも処理済みデータとも一致しない場合の処理

注目している大ブロックでドット配列がディザパターンと一致せず，処理済みのデータ配列とも一致しなかった場合，ブロックをさらに分割して得られる小ブロックにてディザパターンとの一致を評価する。

ディザパターンに一致する小ブロックについては，そのブロックの代表値と直前のブロックの代表値との差分を，一致しないブロックについてはデータを2画素分まとめて1バイトデータとしてそれぞれ別のハフマン符号化を行う。

代表値が見つかった小ブロックについては，3～4ビット程度に圧縮される。すなわち，小ブロックを32ビットとすれば，概ね1/10程度に圧縮される事になる。

一致しないブロックについては元のデータの1/2程度に圧縮される。ここではハフマンコードは頻度分布を基にあらかじめ作成したものを用いた。

2. 4　実際の符号化

ある行の先頭部分を符号化した例をFig.5 に示す。最初に画像を大ブロックに分割する。代表値の初期値を0とし，最初のブロックの代表値が108であるとすると，代表値差分は108となる。これをコード1にてハフマン符号化し，“111001101100”と表現する。

次のブロックも代表値で符号化する，差分は0なので，0をコード1にてハフマン符号化し“0”と表現する。

次のブロックではブロックが4分割されている。ブロック分割制御信号“260”を同じくコード1でハフマン符号化する。

その後，どの小ブロックが代表値表現であるかのフラグが4ビット，さらに対応する代表値と，画素値の符号が続く。

小ブロックの代表値差分はコード2で，画素値はコード3でハフマン符号化する。大ブロックの代表値差分と小ブロックの代表値差分は別々のハフマンコードで符号化するが，差分を求める場合は両者を区別せず直前の値を用いる。

過去のデータとの一致が見つかったときは，一致位置を符号化する。まずコード1で過去との一致を示す制御信号260を符号化する。つぎに，一致位置が直前に算出された位置データと一致したかどうかのフラグを書く。ここでは最初の位置データなので，不一致の“0”を出力し，続いて位置データの2バイトを固定長8ビットで符号化する。2であるから”00000010“である。次の大ブロックでは代表値差分が141であるので，141をコード1で符号化する。以下同様の符号化を画像の右端まで繰り返す。

3種類のハフマンコードとフラグ，固定長整数が混在する符号化であるが，各々の大ブロックの符号は必ずコード1で始まり，大ブロック内の符号化をどの様に行うかまで示しているので，混乱なく展開処理することができる。

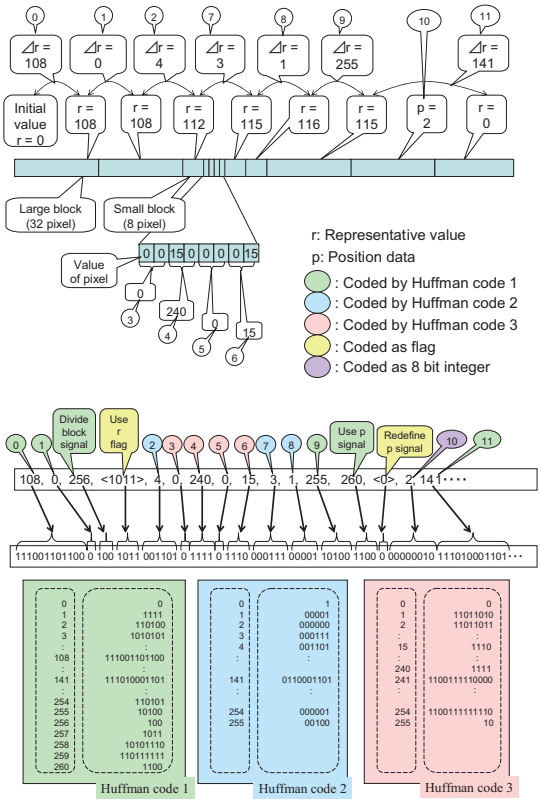

Fig.5 Coding in CDI

CDIは，画像の種類にほとんど関係なく，符号化すべき信号の頻度分布が変化しない特徴がある。よって，あらかじめ設定した固定ハフマン符号化で安定した高性能圧縮が実現できる。

2. 5　性能比較

社内で画像評価に用いられている81の標準画像について4bitハーフトーン処理を行い，従来手法との性能比較を行った。評価対象として，3種類の従来手法A，B，Cに対して，比較実験を行った。但し，従来手法Cは多ビット対応が難しいので，600×2400dpiの2値ハーフトーン処理にて代用した。これらの結果をTable 1 に示す。

圧縮率は，（圧縮後のデータサイズ）/（圧縮前のデータサイズ）で定義し，パーセント表記した。圧縮性能が優れているほど小さな値となる。

この表より，CDIは，いずれの圧縮手法と比較しても，高い平均圧縮性能を実現でき，かつ，画像間のばらつきが減少し，メモリの有効利用が可能になることが判る。

Table 1 Comparison of compression ratios (%)

	CDI	Method A	Method B	Method C
Avg.	4.0%	5.0%	4.3%	6.4%
SD	3.0%	3.7%	7.7%	5.3%
Worst	21.5%	23.21%	83.8%	37.3%

2. 6　回路規模比較

回路設計を行い回路規模について，ゲート数換算で，従来手法Aとの比較を行った。Table 2 に示すように，回路規模が概ね1/3に削減された。

Table 2 Comparison of sizes of compression circuits (k gates)

CDI	Method A
207	584

3　結論

プリンターのコストと性能を両立させるため，ディザパターンを利用した新しい可逆圧縮技術を開発した。この結果，従来技術に比べ，圧縮性能が20%向上し，また回路規模が概ね1/3になった．本技術は，今後，プリンター及びこれらをベースにしたプリンター複合機に搭載予定である。

●参考文献

1）M. ネルソン/J.-L.ゲィリー , データ圧縮ハンドブック [改定第2版]

2）David Salomon, Data compression the complete reference 3rd edition