FineType：カラー液晶の特徴を利用した高精細文字表示技術

手塚忠則† 田路文平† 吉田裕之†

我々はデバイスの変更なしにより高精細な文字を表示する技術である FineType を開発した．FineType は，カラー液晶の R，G，B の各発光素子が規則的に配置されていることに着目し，発光素子の並ぶ方向に対して約 3 倍の解像度の向上を実現する．これにより，特に斜めの線などの部分をスムージングし，通常の表示に比べてより見やすい文字表示を実現する．本稿では，FineType の特徴的な部分である，ビットマップフォントから 3 倍解像度のフォントを生成する部分と，文字表示時に発生する色ずれ抑制処理について述べ，評価実験の結果を示す．この結果，FineType を利用することで通常の表示に比べて視覚的に見やすい表示が行えることが確認できた．

FineType: Hi-resolution Font Rendering Technology Utilizing a Color LCD Device

Tadanori Tezuka,† Bunpei Toji† and Hiroyuki Yoshida†

We are studying and developing sub-pixel rendering technology “FineType” that realizing the hi-resolution text rendering without changing devices. FineType uses the feature of a color LCD that its RGB emit light (called sub-pixel) is regularly located in a row and realizes about three times higher display resolution of text drawing. In this paper, we describe the following feature of FineType text rendering technology, (1) generating the three times higher resolution font from normal bitmap font, (2) preventing the color fringe by sub-pixel drawing, and shows the experimental result.

1. はじめに

近年，携帯電話や PDA を利用してメールを読んだり，ウェブを閲覧したりするなどといった，小型の端末で「文字を読む」ことが広く行われるようになった．また，電子ブックのような携帯型の読書端末も提案されており，今後は小型の携帯端末で文字を読む頻度は高くなると考えられる．

これらの携帯端末の多くは表示デバイスとして主にカラー液晶が利用しているが，液晶の解像度はけして高いとはいえず，文字を表示するとドットが見えてしまい，全体的にギザギザした感じになってしまっていた．

この問題を，液晶デバイスを変更せずに解決する手段として，カラー液晶のデバイスの特徴を利用して解像度感を高めるサブピクセルレンダリング技術がある[1),2)]．これは，カラー液晶の R，G，B の発光素子が規則的に配置されていることを利用し，発光素子の並びの方向に対して解像度を向上させる技術である．この技術を利用したものとしては，Microsoft の ClearType[3)~5)]，Adobe Systems の CoolType[6),7)] がある．ClearType と CoolType は，アウトラインフォントをビットマップに展開するときに水平方向にオーバサンプリングし，これを液晶の発光素子に割り付けることで高解像度表示を実現する．しかし，アウトラインフォントの展開は CPU の能力が低い PHS や携帯電話などの低性能な携帯端末で利用するには処理が重すぎるという問題がある．

これを解決する手段としては，あらかじめサブピクセルレンダリングされたフォントを用意しておくという方法が考えられる．しかし，サブピクセルレンダリングされたフォントデータは，多値データ（R，G，B 各 8 ビットなど）となるため，2 値のビットマップフォントに比べてかなり大きくなってしまう．搭載するメモリ量が少ない端末では，このデータ量の増加が問題となる．

そこで，我々は既存のビットマップフォントから動的に 3 倍解像度のデータを生成し，サブピクセルレンダリングを行う FineType を開発した[8),9)]．この技術では，比較的軽い処理でサブピクセルレンダリングを行うため，CPU 能力が低く，搭載メモリが限られる

† 松下電器産業株式会社マルチメディア開発センター
Multimedia Development Center, Matsushita Electric Industrial Co., Ltd.

低性能な携帯端末で高速なサブピクセルレンダリングを実現する．本稿では，サブピクセルレンダリング技術の概要を説明し，FineType の特徴である，(1) 動的な 3 倍解像度フォント生成，(2) 色にじみの少ないフィルタ処理について述べ，評価実験の結果を示す．

2. サブピクセルレンダリング技術

カラー液晶，プラズマディスプレイでは，**図 1** のように R，G，B の発光素子（以下，サブピクセルと呼ぶ）が規則正しく配置されている．現在利用されている多くの液晶は，図 1 のように水平方向に R，G，B のサブピクセルが並び，R，G，B の 3 つで 1 つの画素（ピクセル）を構成する．サブピクセルレンダリング技術では，この R，G，B の各サブピクセルをそれぞれ異なる画素として考え，それぞれのサブピクセルに異なる画素のデータを割り付ける．たとえば，解像度が 240×320 の液晶デバイスは，サブピクセルを個別の画素と考えると 720×320 の解像度の液晶デバイスと考えることができる．このようにサブピクセルを個別の画素と考えることで，3 倍の解像度を持つデバイスと考えることが可能になる．

次に，文字を表示する場合について説明する．たとえば，7×7 ピクセルの範囲に「A」という文字を表示する例を考えてみる．通常は，**図 2** (a) のように 7×7 ドットの A を表示する．しかし，サブピクセルを単位として考えると，7×7 ピクセルの範囲には 21×7 のドットが表現できることになるので，図 2 (b) の A が表示できる．

しかし，図 2 (b) をそのまま表示した場合には色ずれの問題が発生する．ここでいう色ずれとは，本来は黒い文字を表示したいのに，赤や青などの色が文字中に見えてしまうというものである．これは，R，G，B のサブピクセルがそれぞれ異なる色であり，サブピクセルレンダリングにより R，G，B の発光バランスが崩れた部分に色が見えてしまうことが原因である．**図 3** (a) は，図 2 (b) の A をそのまま表示した場合に各ピクセルが何色になるかを示したものである．このように，R，G，B の発光が不均一な部分には「黄色（Y）」や「シアン（C）」などの色が見えてしまう．

この色ずれを抑制するために，サブピクセルレンダリングでは，図 2 (b) のデータに対してローパスフィルタ処理を施し，輝度のばらつきを抑える．ローパスフィルタにより，変化の激しい文字のエッジ部分が平坦化され，結果として色ずれが抑制される（図 3 (b)）．しかし，平坦化の度合いが強すぎると，エッジ部のぼけ味が増加し，読みにくくなってしまう．このため，

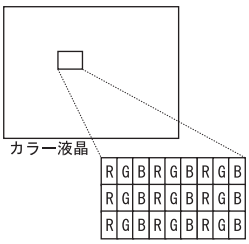


図 1 カラー液晶のサブピクセルの配置
Fig. 1 The alignment of the color LCD sub-pixels.

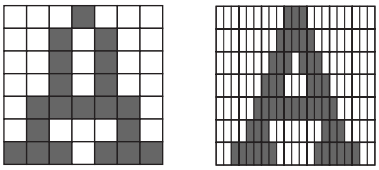


図 2 ピクセル精度とサブピクセル精度
Fig. 2 Pixel accuracy and sub-pixel accuracy.

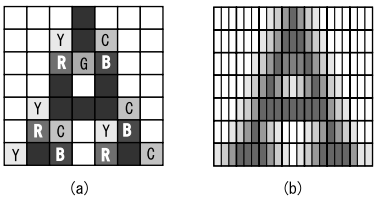


図 3 色ずれの発生とフィルタ処理
Fig. 3 The before and after example of filtering for preventing color fringe.

フィルタ係数はぼけ味と色ずれのバランスを考えて決定する必要がある．

以上のように，文字をサブピクセルレンダリングを用いて高精細に表示する場合には，(1) 3 倍解像度の文字の生成と，(2) 色ずれ防止処理の 2 つを行う必要がある．

ClearType や CoolType などでは，この (1) の 3 倍解像度の文字の生成はベクトルフォントのラスタライズの倍率を変化させて実現している．この方法では，必要な解像度のビットマップを用意することができるが，ベクトルフォントのラスタライズは処理量が大きく，低性能な携帯端末では処理コストが大きい．これを解決するために，ClearType では，フォントのキャッシュを利用することで，ラスタライズの回数を削減している．しかし，フォントキャッシュは，HDD などの記憶装置を持たない携帯電話などでは，メモリを大きく圧迫してしまうため，利用することが難しい．

Gibson ら[1] の方式では，(2) の色ずれ抑制処理として，注目サブピクセルと左右隣接 2 サブピクセルの計

5 サブピクセルに対して 1 : 2 : 3 : 2 : 1 の比率のフィルタを施す．この方法では，色ずれは抑制されるもののフィルタによるボケ味が大きいという問題がある．

我々は，携帯電話をターゲットとし，既存のビットマップフォントから動的に 3 倍解像度のフォントを生成するという手法をとった．これにより，ベクトルフォントの展開に比べて処理量を削減し，加えて，使用メモリ量も削減した．

また，色ずれ防止処理でカラー液晶のサブピクセル R，G，B の輝度特性を考慮したフィルタ設計を行うことで，色ずれを抑えつつ，フィルタによるボケ味を抑えるようにした．さらに，フィルタ処理をテーブル参照で行うようにすることで，フィルタ演算処理の処理量を抑えている．

3. FineType

3.1 概　　要

図 4 に FineType の基本的な処理の流れを示す．入力された文字列は，まずビットマップフォントを参照してビットマップデータに変換される．ここで変換された文字列のビットマップは，表示するサイズと等倍のデータであり，次にこれをもとに 3 倍解像度のデータを生成する (1)．この処理を行うのが 3 倍解像度データ生成処理で，ここでは，注目するピクセルの周辺 8 ピクセル（中心を除く 3×3 の領域）の状態を見ながらスムージングされた 3 倍解像度のビットマップデータを生成する．スムージング処理で生成した 3 倍解像度のビットマップデータには，色にじみを抑制するためのフィルタ処理が行われ (2)，その結果をカラー液晶に表示する．

FineType では，上記処理を低能力の携帯端末で十分実行可能なソフトウェアとして実装した．

3.2 3 倍解像度データ生成

3.2.1 テーブル参照によるスムージング処理

3 倍解像度ビットマップ生成処理を**図 5** に示す．図 5 のように，オリジナルのビットマップフォント（図 5 (1)）は文字部が 1（図では黒いピクセル），背景部が 0（図では白いピクセル）によって表現されているとする．本技術では，まず，3 倍解像度のビットマップを格納するメモリを 0 クリアしておく．次に，オリジナルのビットマップの黒いピクセルを順次操作により検索し，黒いピクセルを見つけた場合には，そのピクセルを注目ピクセルとしてその周辺 8 ピクセルのデータを取得し，これをもとに図中 (2) のようなアドレスを生成する．このアドレスを用いてあらかじめ用意したテーブルを参照し，7 ビットのデータを得る．

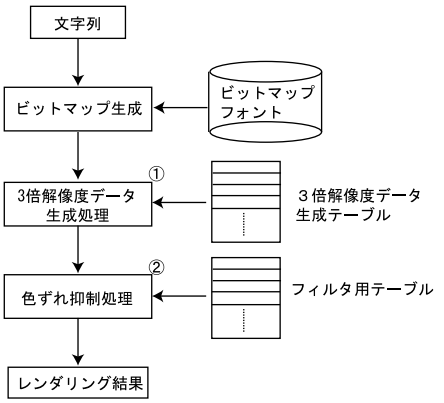


図 4 FineType の処理フロー
Fig. 4 A processing flow of FineType.

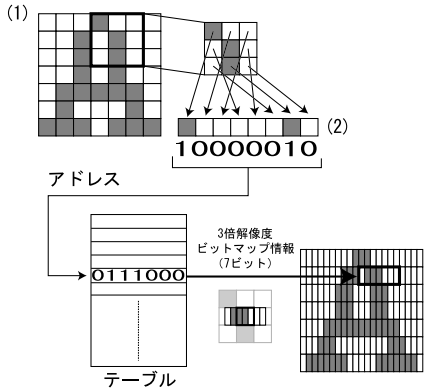


図 5 3 倍解像度ビットマップ生成処理
Fig. 5 A process for creating 3 times higher resolution bitmap.

この 7 ビットは，注目ピクセルを 3 倍にした場合のピクセル中の 3 つのサブピクセルの値と，注目ピクセルの左右 2 サブピクセルの値である．この 7 ビットを参照しながら，ビットの値が 1 である部分について 3 倍解像度のビットマップの対応部分を黒 (1) に書き換える．

本技術では，この書き換えにより，スムージングのための 2 つの操作を実現する．1 つは，注目ピクセルを左右に 1 サブピクセルだけずらすような「シフト操作」であり，1 つは 1 サブピクセルだけ足す「拡張操作」である．

シフト操作　たとえば，注目ピクセルと周辺 8 ピクセルの関係が**図 6** (a) である場合，テーブル参照により出力されるデータは 0001110 となる．これは，注目ピクセルの画素を右に 1 サブピクセルだけシフトさせたものとなり，これをシフト操作と呼ぶ．この操作により，上下ピクセルと注目ピクセルのサブピクセル精度のスムーズなつながりを

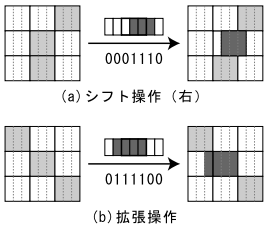


図 6 シフト操作と拡張操作の例
Fig. 6 Examples of shifting and expanding operations.

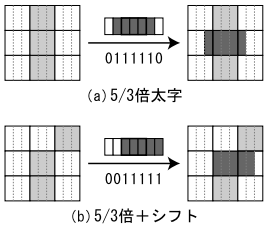


図 7 サブピクセルを使った太字処理の例
Fig. 7 Examples of sub-pixel bold pattern.

実現できる．

拡張操作 図 6 (b) のようなパターンの場合，液晶のようにピクセルがはっきりと表示されるデバイスでは，個々のピクセルの間に隙間が空いているように見えてしまう．そこで，1 ピクセルだけ重ねることで，隙間が空いているように見えるのを防ぐ．これの操作を拡張操作と呼び，これを行うことで図 6 (b) のようなパターンの部分をスムーズに見せることができる．

以上のように，周辺 8 ピクセルから注目ピクセルとその周辺のサブピクセルのデータを決定することにより，ピクセル境界を越えた変更を可能とした．これにより，シフト操作と拡張操作を実現した．また，黒いピクセルのみ処理を行うことで，背景部（白いピクセル）を含めた処理に比べて処理量を削減した．加えて，テーブル参照のみでスムージング処理を行うことで高速な処理を実現している．

3.2.2 太字の実現

1 サブピクセル単位の拡張処理が行えるのを利用すれば，従来にないフォントの太字化を行うことも可能である．たとえば，16 ドットのフォントなどの場合，1 ピクセルの線幅の文字では，文字サイズに比べて線が細くなってしまい，見にくくなってしまう．逆に 2 ピクセル幅の太字にすると線が太すぎて文字がつぶれてしまう．FineType では，サブピクセル単位で文字の太さを変化させることで，つぶれの少ない太字表示を実現する．

これは，先ほど説明したテーブルの書き換えにより実現可能である．**図 7** (a) は 2 サブピクセルだけ太字にする場合のテーブルの例であり，図 7 (b) は，2 サブピクセル太字とシフト処理を組み合わせた場合の例である．このように，サブピクセル単位の太字とスムージング処理を組み合わせることで，スムージングされた太字を得ることができる．なお，この処理はテーブルを入れ替えるだけで実現できるため，スムージング処理と太字処理のテーブルを個別に用意しておき，変換時に入れ替えて利用することで，スムージングのみと太字のフォントを切り替えることが可能である．この場合，必要なメモリ量の増加は，テーブルごとに 256×7 ビットである．

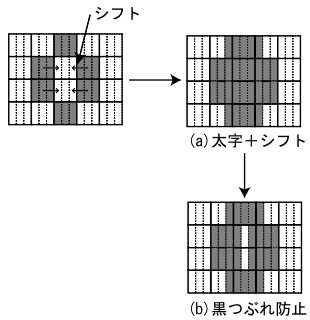


図 8 黒つぶれ防止処理の例
Fig. 8 Examples of buried character.

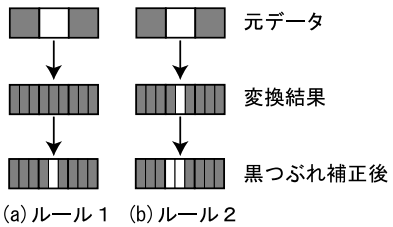


図 9 黒つぶれ防止処理を行うパターン
Fig. 9 The patterns of process for avoiding buried character.

3.2.3 黒つぶれ防止処理

前述した太字＋スムージングを行った場合，フォントによっては黒つぶれが発生する場合がある．**図 8** (a) は黒つぶれが発生するデータの例である．このようにスムージングにより左右からシフトが行われた場合は，本来白であったピクセルがつぶれてしまう．

これを解決するために，スムージング処理では，テーブル参照による 3 倍解像度データ作成後，オリジナルデータのピクセルが白の部分について走査を行い，白の部分が**図 9** のパターンになった部分について，図のような変換操作を行う．これにより黒つぶれを防止することができる．

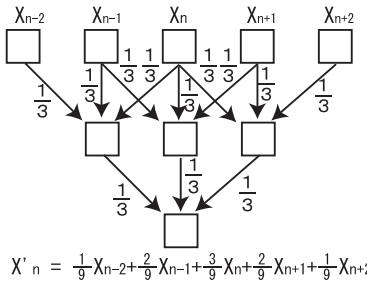


図 10 フィルタ係数（Gibson の方式）
Fig. 10 A filter coefficient (Gibson's).

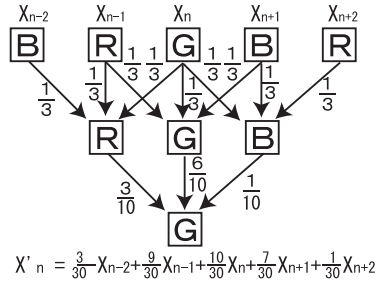


図 11 フィルタ係数（FineType，緑用）
Fig. 11 A filter coefficient (FineType, for green subpixel).

なお，図 9 (b) のような，中央のサブピクセルみが白のものについて，左側のサブピクセルを白に戻すことで，スムージングのみの場合の，「ぱ」などの丸の部分のつぶれ感を防止することができる．この場合，左右でピクセル数が不均等になるが，次のフィルタ処理によりぼかしが行われるため，不均等感は削減される．

このように黒つぶれ防止処理を行うことで，先に示した図 8 (a) のデータを図 8 (b) のように変化させることができる．

3.3 フィルタ処理

3.3.1 色ずれ抑制のためのフィルタ設計

前述したように，Gibson の方式では，色ずれ防止フィルタとして，1:2:3:2:1 の比からなるフィルタを利用していた．これは，**図 10** のように，周辺 5 サブピクセルに対して 1/3 の係数のフィルタを 2 回適用した場合のフィルタ係数となる．

液晶のサブピクセルを考えた場合，各サブピクセルは R，G，B の発光素子であり，それぞれ輝度（明るさ）の度合いは同じではない．Gibson の方式ではこの点について考慮しておらず，3 色のサブピクセルを等価なものとして扱っていた．

本研究では，R，G，B の各色の輝度貢献度が異なるという点に着目し，輝度貢献度を加味したフィルタの設計を行い，これにより輪郭のボケ味を抑え，かつ色ずれを減らした[8]．

具体的には，1 段目のフィルタに 1/3 のフィルタを施し，2 段目で R，G，B の輝度貢献度を加味したフィルタを施す．これにより，輝度貢献度を加味したフィルタ処理を行う．**図 11** は緑（G）のサブピクセルのフィルタ係数の例である．この例では，輝度貢献度を R ： G ： B = 3 ： 6 ： 1（RGB → YCbCr 変換式より）としているが，実際には液晶の特性を測定した結果を利用する．なお，図中の Xn は緑のサブピクセルを表しており，そのサブピクセルの左右に隣接するサブピクセルを Xn-2，Xn-1，Xn+1，Xn+2 と表現している．

3.3.2 テーブル参照によるフィルタ処理

図 11 で示されるフィルタ処理を行う場合，5 回の積計算と，4 回の和計算が必要になる．これをサブピクセルごとに行うのは処理能力の低い携帯端末では大きな負荷となってしまう．

そこで，本研究では，3 倍解像度のビットマップデータが 2 値データであることを利用し，あらかじめ計算しておいたフィルタ演算結果をテーブル参照するだけでフィルタ計算が完了できるようにした．これにより，積和演算をサブピクセルごとに繰り返すのに比べて大幅な処理量の削減を行った．

2 値のデータに対し 5 タップのフィルタを行う場合，それぞれのビットは 0，1 のどちらかしかとらないので，フィルタ演算結果のとりうる値は 5 ビット（32 通り）となる．つまり，32 通りのデータをあらかじめ計算してテーブル化しておけば，後はテーブルデータを参照するだけでフィルタ計算を完了できる．なお，フィルタは R，G，B それぞれで異なる係数を利用するため，実際のテーブルサイズは $32 \times 3 = 96$ エントリとなる．

本研究では，さらに処理速度を向上させるために，ピクセル単位（R，G，B の 3 個のサブピクセル単位）でフィルタ処理を行うようにした．この場合，7 サブピクセルを見ることで R，G，B のデータを一括して生成する．なお，この場合は，128 エントリのテーブルが必要になる．

テーブル参照による色ずれ抑制処理の概要を**図 12** に示す．まず，3 倍解像度のビットマップデータの注目ピクセルのデータとその左右 2 サブピクセル分のデータの計 7 ビットを取り出す．次に取り出した 7 ビットのデータを利用してテーブルを参照し，フィルタ後の R，G，B の値を取り出す（テーブルには事前に計算した値が格納されている）．次に，取り出した値をカラー液晶の対応するピクセルに出力する．

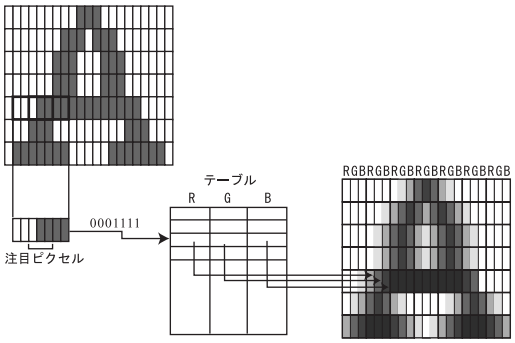


図 12 テーブル参照によるフィルタ処理
Fig. 12 A filter processing utilizing table reference.

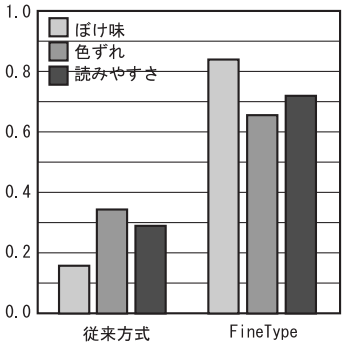


図 13 フィルタ処理の主観評価結果
Fig. 13 The result of subjective evaluation of filter processing.

4. 評価実験

4.1 フィルタ処理

フィルタ係数に R，G，B サブピクセルの発光素子の輝度への貢献度の違いを反映したことによる効果を確認するために，Gibson の方式（1:2:3:2:1）でフィルタ処理を行った場合と，輝度貢献度を加味してフィルタ処理を行った場合について，主観評価および客観的な指標による評価を行った．以下，評価結果を示す．

4.1.1 主観評価結果実験

主観評価では，2 つのフィルタ係数の組合せを提示し，一対比較によりどちらが優れているかを合計 16 人に答えてもらった．優れている場合には 1 点，同じ場合には 0.5 点を与え，得点を比較した．

なお，評価項目は以下のとおりである．

- 文字の読みやすさ
- ボケ味の少なさ
- 色ずれの少なさ

結果を**図 13** に示す．主観評価の結果，文字の読みやすさ，ボケ味の少なさ，色ずれの少なさのそれぞれについて輝度貢献度を加味したもののほうが良いという結果が得られた．特に，ボケ味については，両者の間に大きな差が出た．

4.1.2 客観的な指標による評価結果実験

色にじみとボケ味という点について客観的な評価を行うために，以下の指標を用いて評価実験を行った．

画像品質誤差 PSNR（Peak Signal to Noise Ratio） PSNR は広く画像品質評価の尺度として利用されている[10]．ここでは，サブピクセル単位の PSNR を以下の式により求めた．

$PSNR = 10 \log_{10} \frac{MAX^2}{MSE} [db]$

$MAX =$ 画像の最大輝度

$MSE = \frac{\sum (x_i - \hat{a})^2}{M}$

$M =$ 総サブピクセル数

色ずれを無視すればフィルタ処理を行わないサブピクセル精度のレンダリング画像が理想的な画像であるので，これを理想画像としフィルタ処理による変化を求めた．なお，この数値が大きくなるほど理想的な表示に近いことを示す．

メトリック彩度，メトリック明度差 知覚的にほぼ均等な歩度を持つ色空間（均等色空間）CIE L*a*b* 色空間でのメトリック彩度とメトリック明度である．メトリック彩度は $\sqrt{(a*^2 + b*^2)}$，メトリック明度は L*と定義される．

メトリック彩度は，色の鮮やかさを示す尺度であり，これが 0 の場合は無彩色を意味する．したがって，これが 0 に近づくほど色ずれが小さくなると考えることができる．メトリック明度は明るさを示す尺度で，無彩色の場合は 0 ならば黒，100 ならば白を示す．本実験では，フィルタ処理前のデータを理想として，これとフィルタ処理後の明度のずれであるメトリック明度の差を求めた．この差は，フィルタによる平坦化の度合いを示すので，これが大きい場合はボケ味が強いとした．

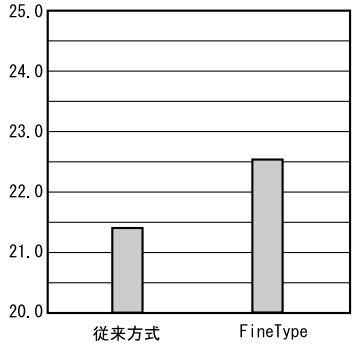
以上の指標により比較した結果が，**図 14** のグラグである．図より，輝度貢献度を考慮したフィルタ処理の方が，PSNR が大きく，メトリック彩度，明度差の双方とも小さいことが確認できた．

これはつまり，客観的な指標による評価でも輝度貢献度を考慮することで，理想の表示に近く，色ずれ，ボケ味ともに少なくなっていることを示している．

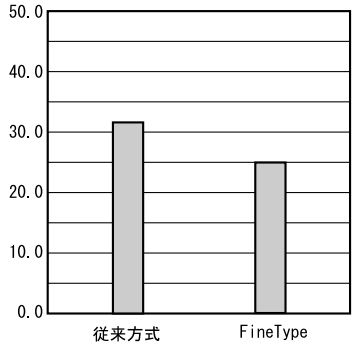
4.2 3 倍拡大処理

4.2.1 パターンの出現頻度の調査

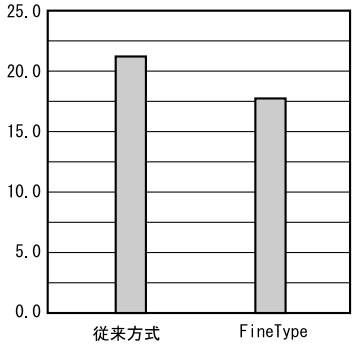
ここでは，3 倍解像度のデータ作成時のスムージング処理がどれくらいの部分に適用されているかを調査した．利用したテキストは夏目漱石の「坊ちゃん」全



(a) PSNR



(b) メトリック彩度



(c) メトリック明度差

図 14 客観指標による比較
Fig. 14 Comparison with objective indexes.

文（約 10 万文字）で，フォントの種類によるスムージングの度合いの差をあわせて調査するために明朝（MS 明朝，9 pt）とゴシック（MS ゴシック，9 pt）の PC で利用される代表的な 2 書体を用いた．

結果を**表 1** に示す．表を見るとどちらの書体の場合も文字（黒い部分）の割合が全体の約 20%であることが分かる．本技術では，注目ピクセルが「黒」の場合のみ，周辺 8 ピクセルを利用したテーブル参照を行うが，約 80%の部分についてはこの処理を省くことができていることが分かる．したがって，全体に対して処理を行うのに比べて処理量を約 1/5 に削減できていることが確認できた．

表 1 「坊ちゃん」でのパターン出現頻度
Table 1 Frequency of an appearance of the pattern in a novel.

	MS ゴシック	MS 明朝
文字（黒い部分の割合）	20.00%	20.42%
スムージングされるパターンの占める割合	24.32%	27.47%
上位 20 パターンの出現頻度合計	67.88%	54.55%
上位 20 パターン中のスムージングされるパターン数	7 パターン	8 パターン

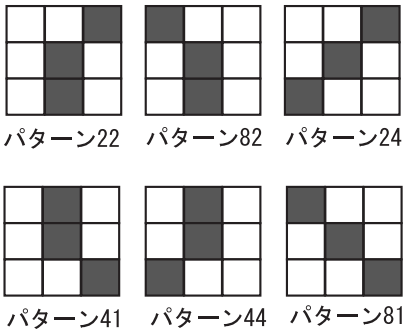


図 15 2 書体に共通する上位のスムージングパターン
Fig. 15 The smoothing pattern of the higher rank common to two typefaces.

また，処理した部分のうちスムージング対象になった割合はどちらのフォントでも約 25%であることが分かる．これは，文字を構成する画素のうち約 1/4 に対してスムージング処理が施されたことを示している．さらに，周辺 8 ピクセルのパターン（全 256 パターン）のうち上位 20 パターンで MS ゴシックで 67%，MS 明朝で 54%であること，上位 20 パターンのうち MS ゴシックで 7 パターン，MS 明朝で 8 パターンがスムージング対象となったことが確認できた．

なお，上位 20 パターン中に 2 書体に共通するスムージングパターンは**図 15** の 6 つである．

4.2.2 主観評価実験

主観評価では，FineType で高精細表示されたテキストが通常表示にくらべ読みやすく感じるかという点と，スムージングの度合いの違いが主観評価に現れるのかという点の 2 点について調査を行った．被験者に提示したテキストは，**図 16** の 2 種類（カレンダー，サンプルテキスト）で，それぞれ表 2 のようなパターン頻度である．**表 2** のように，カレンダーの方がスムージングされる割合が大きく，サンプルテキストに比べて読みやすさの向上が期待される．

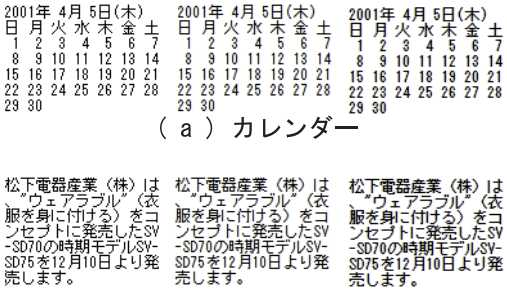


図 16 主観評価に利用したテキスト
Fig. 16 The text utilized for subjective evaluation.

表 2 主観評価テキストでの出現頻度
Table 2 Frequency of appearance in the subjective evaluation text.

	カレンダー	サンプルテキスト
文字（黒い部分の割合）	13.84%	20.49%
スムージングされるパターンの占める割合	26.16%	20.41%

実験では，この 2 つのテキストを，通常表示（サブピクセルレンダリング処理を行っていないもの）とFineType2 種（スムージング，1.33 倍太字）提示し，最も読みやすいと思うものを 1 つだけ選択してもらった．なお，被験者の数は 50 人である．**図 17** は実験の結果である．実験の結果，カレンダー，サンプルテキストともにスムージング文字が読みやすいと答えた人数が過半数を占めた．また，1.33 倍太字を加えた本技術による表示が良いと答えた人数はカレンダーで 78%，サンプルテキストで 72%となった．以上のように，本技術による表示の方が読みやすいと答えた人数が 70%を超える結果となった．

また，カレンダーとサンプルテキストの 2 つを比較した場合，カレンダーの方が読みやすいと答えた割合が多かった．これは，表 2 で示したようにスムージングが施される割合がカレンダーの方が多いということに起因すると考えられる．

主観評価の結果，通常表示に比べて本技術の表示が読みやすく，また，スムージングが施される割合の多いカレンダーの方がサンプルテキストにくらべて読みやすいと感じた人数が多いことが確認できた．

5. 関 連 技 術

前述したように Microsoft の ClearType，Adobe Systems の CoolType はアウトラインフォントをベースにサブピクセルレンダリングを行う．アウトラインフォントは，スクリーンフォントとして利用されている 12×12 ドット，16×16 ドット程度のフォント表示に利用する場合には，ヒントを利用しても文字が潰れてしまったり，形が崩れてしまったりして読みにくくなることがよく知られている．このため，Windows などでもスクリーンフォントサイズにはビットマップフォントが用いられている．特に日本語はこの傾向が強く，小さなサイズのフォントではアウトラインフォントは文字を正しく表現できない．実際に，WindwosXP 搭載の ClearType でも，Times などのフォントについては小さなサイズからサブピクセルレンダリングを行うが，日本語については，18 ポイント程度のサイズからしか ClearType が動作しない．

一方，我々の方式では，ビットマップフォントをベースにサブピクセルレンダリング処理を行うため，小さなフォントに対してもサブピクセルレンダリングによる高精細描画を実現できる．

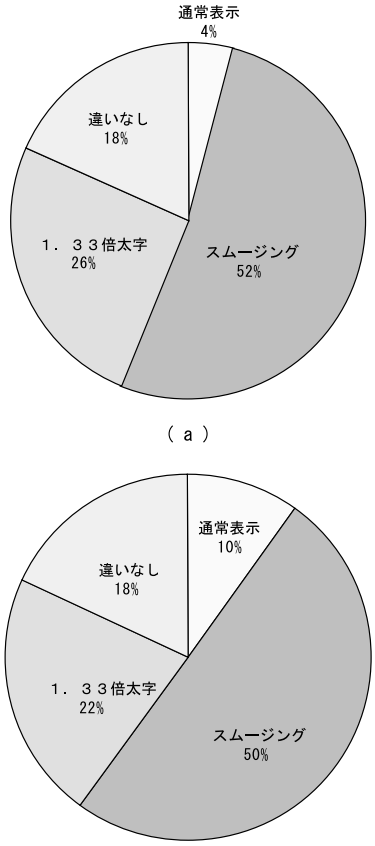


図 17 主観評価結果
Fig. 17 The result of subjectivity evaluation.

6. ま と め

本稿では，携帯電話のように処理能力が低く，リソースの限られる携帯端末向けのサブピクセルレンダリング技術である FineType について述べた．FineType では，ビットマップフォントから 3 倍解像度のデータを作成することで，既存のフォントから高精細な表示を可能にした．また，テーブル参照のみで 3 倍解像度のフォントを生成し，テーブル参照のみでフィルタ処理を行うことでサブピクセルレンダリングに必要な処理量を大幅に削減している．さらに，フィルタ処理に関して，液晶の R，G，B のサブピクセルの輝度貢献度を考慮することで，色にじみとボケ味の削減を図った．

実験の結果，輝度貢献度の加味と 3 倍解像度データ生成時のスムージングの効果により見やすい表示を実現できたことが確認できた．

今後は，本技術を文字だけでなく，グラッフィクス描画にも応用していく予定である．

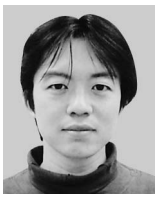
参 考 文 献

1) Gibson, S.: Sub-Pixel Font Rendering Technology. http://grc.com/cleartype.htm
2) 岡田　哲ほか：カラー液晶対応 LC フォント LCFONT.C，シャープ技報第 81 号 (2001).
3) Microsoft. Co. Ltd.: ClearType—Microsoft ClearType information page. http://www.microsoft.com/typography/cleartype/
4) Microsoft. Co. Ltd.: Method and apparatus for diplaying images such as text, US Patent No.6188385.
5) Microsoft. Co. Ltd.: Mapping image data samples to pixel sub-components on a striped display device, US Patent No.6219025.
6) AdobeSystems. Inc.: Adobe CoolType. http://www.adobe.com/products/acrobat/cooltype.htm.
7) AdobeSystems.Inc.: Device dependent rendering of characters, US Patent No.378237.
8) 田路文平，手塚忠則，吉田裕之：高精細表示を可能とするカラー LCD 向けサブピクセルレンダリング方式の一提案，Visual Computing 2001 予稿集，pp.13–18 (2001).
9) 田路文平，手塚忠則，吉田裕之：FineType：カラー液晶の特徴を利用した高精細文字表示技術，Visual Computing 2002 予稿集，pp.13–18 (2002).
10) テレビジョン学会編：画質と音質の評価技術，昭晃堂 (1991).

(平成 14 年 7 月 3 日受付)
(平成 15 年 1 月 7 日採録)

手塚　忠則（正会員）

1968 年生．1993 年九州工業大学大学院情報科学専攻博士前期課程修了．同年松下電器産業株式会社入社．現在，同社マルチメディア開発センター在籍．情報工学博士．分散コンピューティング，画像処理技術の研究開発に従事．



田路　文平

1999 年名古屋大学大学院工学研究科修士課程修了．同年松下電器産業株式会社に入社．現在，同社マルチメディア開発センター在籍，画像処理技術の研究開発に従事．画像電子学会会員．

吉田　裕之（正会員）

1984 年電気通信大学電気通信学部卒業．現在，松下電器産業株式会社マルチメディア開発センターに所属．知的学習支援システム，画像処理技術に関する研究開発に従事．人工知能学会会員．