

携帯情報端末用スケラブルフォントの試作と評価

安本 護[†] 池田 尚志^{††}
中里 西克[†] 富田 義数[†]

移動体通信技術の発展によって、モバイル環境における情報処理の機会が増大している。今後は、携帯電話などの通信機器に電子メールなどの機能を取り込んだ簡便な携帯情報端末の普及が見込まれる。良好なユーザインタフェースの実現には、高品質の文字表示が望ましい。しかし、このような機器において、大量の記憶領域を必要とする従来のフォント手法を利用することは困難である。本論文では、普及タイプの携帯情報端末においても、任意サイズのフォントを高品質に生成可能とする方法を提案する。この手法は、ストロークフォント方式をもとに隣接ストロークの間隔を調整してつぶれを防止する線間補正処理と出力画素数に合わせてストロークの混み具合を調節する複雑度補間処理を適用するものである。本手法を用いれば、高画素サイズにおける正確な字体の再現と低画素サイズで求められる高い判読性を両立可能である。16人の被験者に対する主観テストの結果、人手によって制作したビットマップフォントに匹敵する品質を得られることが確認できた。

A Scalable Font for Mobile Computing Terminals

MAMORU YASUMOTO,[†] TAKASHI IKEDA,^{††} YUUKI NAKASATO[†]
and YOSHIKAZU TOMIDA[†]

This paper proposes a high quality scalable font for mobile terminals in order to improve mobile computing user interfaces. This font not only expresses an accurate glyph in a large size but also makes smaller output legible. We apply the font compensation technique proposed here to the stroke font to accomplish this. The technique consists of two algorithms. The first algorithm creates a space between adjacent strokes that have been blurred together by scale-down operations. The second algorithm interpolates with a high resolution font and a low resolution font to prevent crowding of the strokes. Experiments show that our proposed method can achieve as a good quality font as the man-made bitmap font.

1. はじめに

携帯情報端末の普及や移動体通信技術の発展によって、モバイル環境における情報処理の機会が増大している。今後は、携帯電話などの通信機器に電子メールなどの機能を取り込んだ簡便な携帯情報端末の普及が見込まれる。

携帯情報端末においても高品質の文字表示を備えた良好なユーザインタフェースが望まれる。しかし、大量の記憶領域を必要とする従来のフォント手法を小型・軽量の携帯情報端末に利用することは困難であった。本論文では、普及タイプの携帯情報端末においても任意サイズのフォントを高品質に生成可能とする方法を提案する。

現状の小型携帯情報端末（PDA）では、画素数が240×160ドットから480×240ドット、ドットピッチが0.2～0.3 mm/dotのモノクロ2階調液晶ディスプレイを使用することが多い。一方、電子メールやWWWなどのアプリケーションにおいて情報の一覽性を確保するには、1行あたり20文字以上の漢字表示が望まれる。そこで、搭載される日本語フォントは、縦横の画素数が12ドット程度の低画素ビットマップフォントが中心となっている。

JIS Z 8513規格¹⁾は、視覚表示装置を用いるオフィス作業に関して、以下の実現を要求している。

- (1) 視距離（目から画面までの距離）が300 mm以上であること
 - (2) 漢字の最小文字高が視角25'であること
 - (3) 漢字の文字構成が15画素×16画素以上あること
- 携帯情報端末使用時の視距離は350 mm程度なので、12ドットフォントの文字高は23'～35'となる。したがって、従来の携帯情報端末用フォントは、視距離と

[†] 三洋電機株式会社ハイパーメディア研究所
Hypermedia Research Center, Sanyo Electric Co., Ltd.
^{††} 岐阜大学工学部
Faculty of Engineering, Gifu University

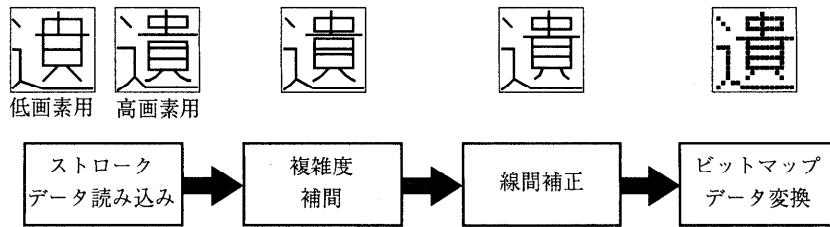


図1 フォント処理全体のながれ

Fig. 1 General flow.

文字高については JIS 規格を満足しているものの、構成画素数は十分といえない。ユーザビリティの向上や字体の正確な表現には、必要に応じて画素数の多い文字を表示できねばならない。また、WWW の記述に用いる HTML に 7 種類の文字サイズ指定が存在するように、アプリケーションによっては文字サイズの任意変更が必要である。

文字サイズを任意に変更可能なスケーラブルフォントとして、アウトラインフォントが実用化されている。しかし、アウトラインフォントは低解像度の画面表示に使用するとつぶれを発生するため、ビットマップフォントとの併用が必要であった。MS-Windows に標準搭載されている明朝体やゴシック体のデータサイズが各々 4~5 MB であるように、アウトラインフォントは大量の記憶領域を必要とする。これに加えてビットマップフォントを使用するのは、大容量記憶装置を持たない簡便な携帯情報端末にとって困難である。

また、欧文アルファベットに比べて字形の複雑な漢字を低画素でつぶれないように出力するには、しばしばストロークの省略が必要になる。しかし、META-FONT²⁾に代表される従来のアウトラインフォントは、このような機構を提供していない。

低画素で可読性の高い文字イメージの生成を試みた研究には、字体を表す骨格情報と書体デザインを表す輪郭情報に文字を分解し、特定の縦線と横線に対してつぶれそうになったら削除してよいという間引き情報を追加するものがある³⁾。しかし、この手法は出力画素数に応じて文字骨格を制御しておらず、出力可能な最小画素数を保証できない。

出力サイズに合わせて最適なデザインのフォントを自動的に作り出す研究には、小さな文字出力用の細字体と大きな文字出力用の特太字体のアウトラインフォントデータを補間し、中間サイズの出力に適した線幅を持つアウトラインフォントを生成するものがある⁴⁾。しかし、この手法は印刷出力を対象とした高解像度向きのものであって、低画素フォントで必要となるストローク省略は考慮されていない。

小型携帯情報端末用フォントに求められる項目を以下に示す。

- (1) 出力画素数を任意に変更可能であること
- (2) 12ドット程度でも文字がつぶれず、判読が容易なこと
- (3) 高画素出力では、省略のない正確な字体を表現できること
- (4) データサイズがコンパクトであること

以上の目標を実現するために、本論文ではストロークフォントに解像度補間処理と線間補正処理を適用することを提案し、試作フォントデータを用いた実験によって、その有効性を示した⁵⁾。

2. フォント処理方式

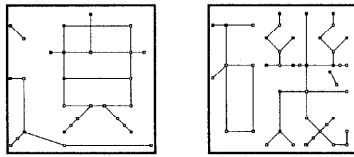
2.1 フォント処理の概要

図1にフォント処理全体の流れを示す。本システムでは、低画素用と高画素用の2種類のストロークフォントデータを入力に用いる。次に低画素フォント生成時に発生する字画のつぶれを防ぐために複雑度補間と線間補正という2つのフォント形状補正処理を適用する。最後にストロークデータから表示用ビットマップデータに変換する。

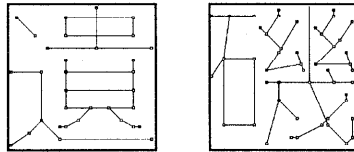
2.2 ストロークフォント方式

ストロークフォント方式は、1辺の大きさ M の正方形枠内にデザインされた文字から骨格形状を特徴付ける点を抽出し、その座標 (X, Y) で文字データを記述する。 X, Y のとりうる最大値 M はメッシュサイズと呼ばれ、これが大きいほど文字形状を高精度に記述できる。ストロークフォントを表示・印刷する際は、特徴点データを任意の出力サイズ S に座標変換し、得られた座標 (X_s, Y_s) を直線または曲線で結んでビットマップデータを得る。

一方、最近のパーソナルコンピュータやプリンタに広く用いられているアウトラインフォント方式は、文字輪郭の特徴点座標を記憶し、表示・印刷の際に特徴点に囲まれた内部をドットで塗りつぶしてビットマップ化するものである。両者を比較すると、高画素にお



(a) 低画素出力用



(b) 高画素出力用

図2 ストロークフォントの例

Fig. 2 Examples of stroke font.

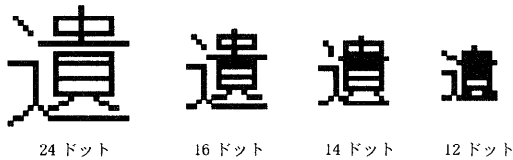


図3 字画つぶれの例

Fig. 3 Examples of stroke blurs.

ける表現力ではアウトラインフォントが優れているが、データサイズや処理速度ではストロークフォントが有利である。すなわち、低画素出力が中心で、しかも十分なハードウェア資源を持たないPDAへ応用するにはストロークフォントが適している。図2にストロークフォントの例を示す。(a)は低画素用フォント D_l 、(b)は高画素用フォント D_h である。

2.3 低画素フォント生成の問題

ストロークフォントやアウトラインフォントを用いれば、任意画素数のビットマップデータを生成可能である。しかし、画素数が極端に少ないと字画につぶれを生じ、判読性が劣化する。これは、欧文と違って字形が複雑な漢字を使用する日本語や中国語では深刻な問題となる。つぶれとは、隣り合うストロークの座標間距離が1となって、ビットマップ化した際に間隙がなくなる現象である。図3は、図2(b)のストロークデータから生成したビットマップデータであるが、16ドット以下ではつぶれによる文字品質の劣化が起きている。字画のつぶれは、次に述べる出力画素数の不足と座標の丸め誤差が原因となって発生する。

2.3.1 出力画素数の不足

字画の混み具合すなわち文字複雑度に対して出力画素数が不足すると、つぶれを発生する。「遺」という漢字の場合、x座標のオーバーラップする水平ストローク

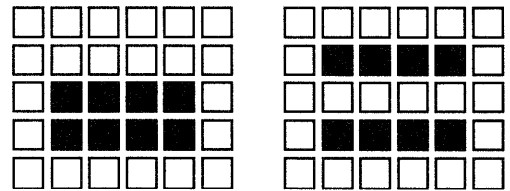
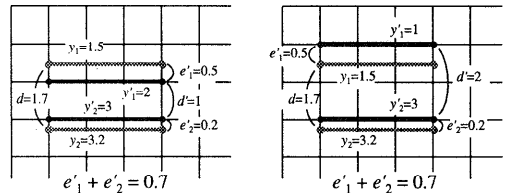
8 愛違遺磯鯛

11 愛違遺磯鯛

16 愛違遺磯鯛

図4 低画素ビットマップフォントの字画省略例

Fig. 4 Examples of stroke abbreviations in low resolution bitmap fonts.



(a) つぶれ発生

(b) つぶれなし

図5 丸め誤差の発生

Fig. 5 Occurrence of rounding error.

は8本である。したがって、水平ストローク間に1画素以上の間隙を確保するには、出力画素数を16以上としなければならない。

グレイスケールフォントを用いれば、ストロークの混み合った部分に中間階調の輝度を割り当ててつぶれを軽減し、判読性を向上することが可能である⁶⁾。しかし、モノクロ2階調ディスプレイではグレイスケールフォントを使用できない。

そこで、文字複雑度以上に低画素のビットマップフォントでは、判読性を低下させないように字画を省略するデザインテクニックが用いられる。図4に人手により制作された低画素ビットマップフォントの例を示す。

2.3.2 座標の丸め誤差

フォントデータの座標変換において少数点以下に端数を生じた場合、これを整数化してビットマップデータに変換しなければならない。このとき発生する丸め誤差の取扱いは、フォントの生成品質に重大な影響を及ぼすことが知られている³⁾。

図5を用いて丸め誤差によるつぶれの発生を説明する。2本の水平ストロークを座標変換した結果、y座標が $y_1 = 1.5$, $y_2 = 3.2$ になったとする。このと

き、ストローク間隔 $d = 1.7$ である。図 5(a) のように両ストロークの y 座標を四捨五入して $y'_1 = 2$, $y'_2 = 3$ を得るとストローク間隔 $d' = 1$ となってつぶれが発生する。ところが、丸め処理前のストローク間隔 $d = 1.7$ の四捨五入結果 $d' = 2$ は字画のつぶれを生じない間隔である。そこで、図 5(b) のように $y'_1 = 1$, $y'_2 = 3$ とする整数化を用いれば、ストローク間隔は 2 となって字画につぶれは発生しない。丸め誤差の累積は、(a), (b) とも 0.7 で等しい。

2.4 フォント形状補正処理

ストロークフォントから低画素フォントを生成する際につぶれ発生を防ぐために、2種類のフォント形状補正処理を提案する。

2.4.1 複雑度補間処理

低画素で読みやすいフォントを得るには、出力サイズに合わせてフォントをデザインするのが有効である。しかし、これでは膨大なデータが必要となってしまう。そこで、式 (1) に示す複雑度補間処理を導入する。

$$X_s = \left(R \cdot \frac{X_h}{M_h} + (1 - R) \cdot \frac{X_l}{M_l} \right) \cdot S$$

$$Y_s = \left(R \cdot \frac{Y_h}{M_h} + (1 - R) \cdot \frac{Y_l}{M_l} \right) \cdot S \quad (1)$$

式 (1) は、出力画素数 S_l に最適化したデザインの低画素出力用ストロークフォント D_l (字画省略あり、メッシュサイズ M_l)、ならびに出力画素数 S_h 以上の使用に最適化した高画素出力用ストロークフォント D_h (字画省略なし、メッシュサイズ M_h) の2つから出力画素数 $S (S_l \leq S \leq S_h)$ のストロークフォント D_s を生成するものである。 (X_l, Y_l) , (X_h, Y_h) , (X_s, Y_s) は、それぞれ D_l , D_h , D_s の座標を示す。

D_l と D_h は、ストローク数およびストロークを構成する点の数が同一となるようにデザインしておく。すなわち、 D_l の字画省略は、複数のストロークを重ねて1本のストロークに見せたものである。

R は D_l と D_h の D_s に及ぼす影響度を決定するパラメータであり、これによって出力画素数に見合った複雑度を D_s に与える。式 (2) を用いると、 S_l と S_h の間を出力画素数 S に応じて均等に内分するパラメータ R が得られる。

$$R = \frac{S - S_l}{S_h - S_l} \quad (2)$$

2.4.2 線間補正処理

図 6 に示す線間補正処理は、ビットマップ化する前のストロークデータにおいて、水平/垂直ストロークの間隔が1とならないように座標を移動し、つぶれを防止する処理である。ストローク間距離 $d = 0$ とな

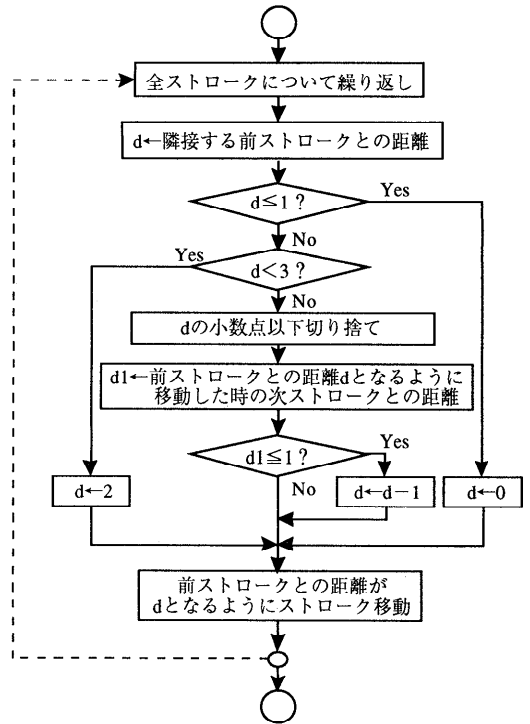


図 6 線間補正処理アルゴリズム
Fig. 6 Stroke space compensating algorithm.

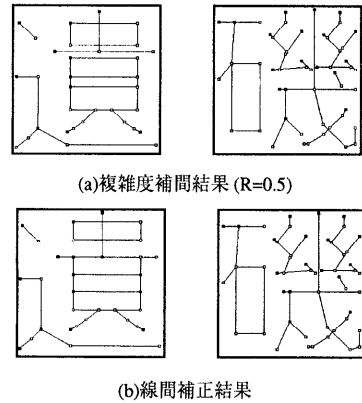


図 7 形状補正の例
Fig. 7 Example of font shape compensating.

たときには、当該ストロークを移動して1つ前のストロークに重ね合わせ、ストロークを省略したように見える。

図 7(a) は、図 2 のストロークデータを $R = 0.5$ で複雑度補間した結果である。「磯」の各ストロークは適切な間隔を保っているが、「遺」の水平ストロークは間隔が不揃いである。低画素のビットマップデータに変換すると、間隔の狭いところがつぶれてしまい、高品質の生成結果とならない。そこで、これを画素数 16

ドットのビットマップ用に線間補正すると図7(b)のようになる。「遺」の水平ストロークには省略と間隔の均一化が行われており、線間補正の効果が分かる。一方、斜めストロークの多い「磯」は、線間補正がはたらかないため変化は見られない。このように複雑度補間と線間補正を行うことによって、出力画素数にふさわしいフォント形状とつぶれの防止を得ることができる。

なお、出力画素数に適したフォント形状となる保証はないが、複雑度補間処理を行わず線間補正処理のみをフォントデータに適用することも可能である。

3. ストロークフォントデータの試作

本論文で提案するフォント方式の有効性を検証するために、情報交換用漢字符号 JIS X 0208 規格の 16 区に収録された漢字から比較的字体の複雑な漢字 14 文字(愛, 逢, 葵, 穠, 鯨, 綾, 慰, 萎, 謂, 違, 遺, 磯, 鯛, 胤)を選び、ストロークフォントデータ D_l と D_h を試作した。制作にあたっては、同一ゴシック体字母をもとに人手によって制作したビットマップフォントの骨格をトレースした。 D_l の制作で参照したビットマップフォントは、画素数が 11 ドットで字画の省略を含んでいる。一方、 D_h では画素数 24 ドットで字画省略のないフォントを参照した。すなわち、 $S_l = 11$, $S_h = 24$ である。次に、ストロークフォントのメッシュサイズは、複雑度補間処理を考慮して 11 と 24 の公倍数 264 に選んだ。このように十分大きなメッシュサイズを使用した結果、 D_h は 24 ドット以上の高画素でも品質の高い出力が可能である。

次に、試作結果から JIS X 0208 全文字 (6879 文字) のストロークフォントを制作した場合の大まかなデータサイズ S_s を式 (3) を用いて見積る。

$$S_s = \text{文字あたり平均画数} \\ \times \text{ストロークあたり平均特徴点数} \\ \times \text{特徴点あたりデータサイズ} \\ \times \text{文字数} \quad (3)$$

JIS X 0208 に含まれる漢字 (6355 文字) を調べたところ、平均画数は 12.341 本/文字であった。次に試作フォントのストロークあたり平均特徴点数を求めると、2.7114 点/本であった。また、特徴点のデータは、複雑度補間を行わないときで 1 点につき 3 バイト (x 座標×1, y 座標×1 および制御情報)、複雑度補間を行うときは 5 バイト (x 座標×2, y 座標×2 および制御情報) で表現できる。以上より、ストロークフォントのデータサイズを見積ると、複雑度補間なしのとき 682 K バイト、複雑度補間ありのとき 1136 K バイト

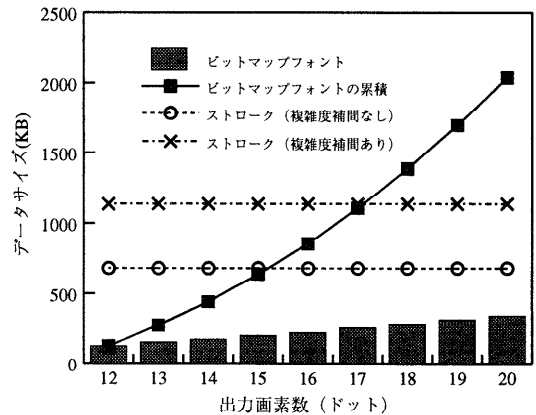


図8 フォントデータのサイズ見積り
Fig. 8 Estimation of font data size.

となった。これは、一般的なアウトラインフォントのサイズが 4~5 M バイトであるのに比べると、かなりコンパクトである。また、ストロークフォント方式では、部首などの組合せによる漢字生成^{3),7),8)}が可能なので、さらにデータ圧縮することも期待できる。

図8にビットマップフォントの画素数ごとのデータサイズ、12 ドットから 1 ドット刻みに複数のビットマップフォントデータを持つ場合の累積データサイズおよび複雑度補間処理の有無による 2 つのストロークフォントのデータサイズを示す。ここで、画素数 n ドットのビットマップフォントのデータサイズ S_b は、式 (4) で与えられる。

$$S_b = \left\lceil \frac{n^2}{8} \right\rceil \times \text{文字数} \quad (4)$$

ただし、 $\lceil \cdot \rceil$ はシーリング関数

ストロークフォントのデータサイズは、複雑度補間なしの場合、画素数 12~15 ドットのビットマップフォントを合計したもので済む。複雑度補間付きの場合でも、12~17 ドットの合計にすぎない。このようにストロークフォント方式によれば、ビットマップフォントを複数種類持つよりも少量のデータで、多様なサイズのフォントを生成可能である。

4. 実験と評価

本論文で提案するストロークフォント方式の効果を評価するために、試作ストロークフォントから次の 6 種類の方法でビットマップデータを生成した。

- G1: D_l (字画省略あり) から生成, 複雑度補間なし, 線間補正なし
- G2: D_l (字画省略あり) から生成, 複雑度補間なし, 線間補正あり

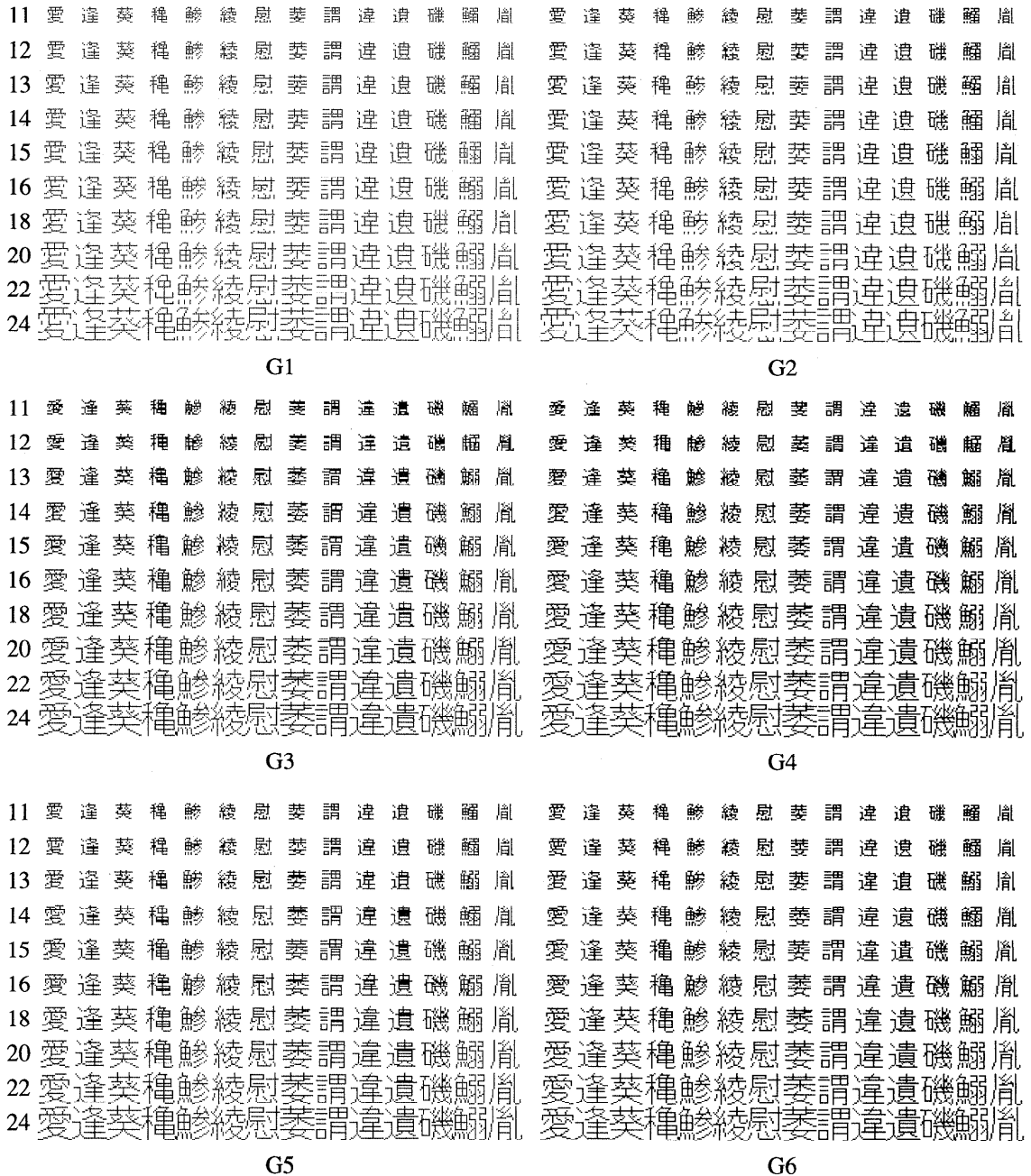


図9 ビットマップデータ生成結果
 Fig. 9 Results of generating bitmap data.

- G3: D_h (字画省略なし) から生成, 複雑度補間なし, 線間補正なし
- G4: D_h (字画省略なし) から生成, 複雑度補間なし, 線間補正あり
- G5: D_l (字画省略あり) と D_h (字画省略なし) から生成, 複雑度補間あり, 線間補正なし

- G6: D_l (字画省略あり) と D_h (字画省略なし) から生成, 複雑度補間あり, 線間補正あり

4.1 生成結果

G1~G6で生成した画素数11ドットから24ドットの結果を図9に示す。また、図10は低画素時の生成品質を明らかにするために、11~16ドット生成結果

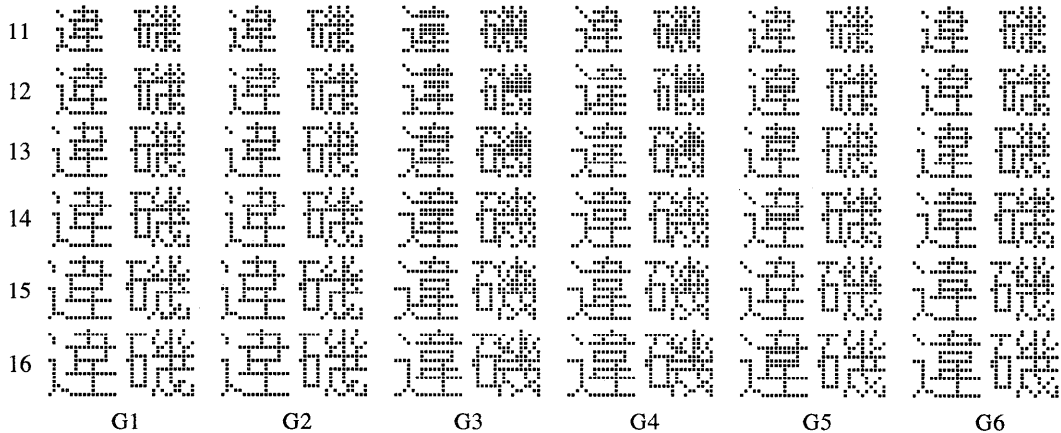


図 10 低画素生成結果の拡大
Fig. 10 Enlargements of generating low-pixel data.

の一部を拡大したものである。

G1は、低画素用のストロークフォントから生成しているの、画素数の小さなところではすっきりとして読みやすいが、大きくなるにつれて字画省略が目立っている。

G2は、G1に線間補正を行ったものであるが、元々G1にはつぶれがないので生成結果はG1と変わらない。

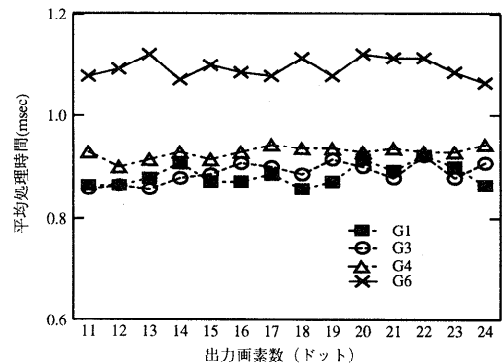
高画素用のデータを用いたG3は、24ドットでは正確な字体を再現できているが、16ドット以下の低画素では、つぶれが目立ち、判読が困難になっている。

G3を線間補正したG4は、水平ストロークの多い「違」や「遺」の低画素展開でつぶれがなく、線間補正による品質の改善が認められる。しかし、出力サイズに合わせた文字骨格の制御は行っていないので、「違」の12、13ドット生成時には文字構成要素の「口」が「一」となっている。このような省略は判読性を低下させ、好ましくない。また、「磯」には線間補正がはたらかないので、G3同様のつぶれがある。

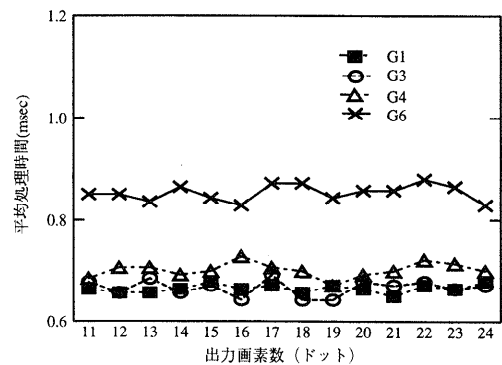
G5は、複雑度補間処理のみで線間補正処理を行わないため、ストローク間隔の乱れによるつぶれが目立ち、判読性が低い。

複雑度補間処理と線間補正処理を併用するG6では、G1～G5に見られる問題も解消し、最小出力画素数の11ドットから24ドットに至るまで判読性の高い結果となっている。

以上、本論文で提案するフォント形状補正処理を行ったG2、G4、G5、G6中、明らかな品質改善が見られたのはG4とG6である。一方、形状補正処理をいっさい使わないG1とG3は、従来のストロークフォント方式の結果を示すものといえる。よって、以降の実



(a) 描画処理を含むとき



(b) 描画処理を含まないとき

図 11 ストロークフォント処理時間

Fig. 11 Processing time of stroke font.

験と評価は、G1、G3、G4、G6を対象に行う。

4.2 フォント処理時間

ストロークフォント処理の内容を機能で分割すると座標計算と描画処理になる。図11は、G1、G3、G4、G6の処理に要した平均時間を、(a)描画処理を含む場合と、(b)座標計算のみで描画処理を含まない場合に分

けて示したものである。各ストロークフォントデータとも、特徴点数は出力画素数にかかわらず一定である。したがって、座標計算のみの処理時間は出力画素数に依存しないはずである。一方、描画を含む処理時間は、出力画素数が大きくなるにつれて増加するであろう。ところが、図 11 (a) は出力画素数によらず、ほぼ一定となった。そこで、出力画素数を 256 ドットまで大きくして、G1 の処理時間を計測したところ、1.2msec となって増加が確かめられた。出力画素数が 11~24 ドットの描画処理には、計測できるほどの時間差はないようである。図 11 (b) については、出力画素数をさらに大きくしても一定であった。

G1, G3, G4, G6 の処理時間を比較すると、形状補正を行わない G1, G3 が最も短く、続いて G4, G6 となった。G1 と G3 のストロークフォントデータは異なるものであるが、特徴点数が同一であるため処理時間にほとんど差が現れないと考えられる。G4 の結果からは、線間補正によって処理時間が G1, G3 よりも約 5% 増加することが分かる。線間補正に加えて複雑度補間も行う G6 は、同じく G1, G3 に比べて約 24% の処理時間増となる。

実験に使用した計算機 (SuperSparcII 75 MHz, 128 Mbyte, Sun OS 4.1.4) では、最も計算量の多い G6 でも約 900 文字/秒の生成が可能であった。最近では、PDA にも数十 MIPS 程度の RISC CPU が搭載されているので、ストロークフォントの利用は十分に可能と考えられる。

4.3 生成品質の評価

被験者 (20 代 7 人, 30 代 9 人, 正常な視力を有する) に対して下記の主観テストを行い, G1, G3, G4, G6 の品質を評価した。

- (1) 評価指標は、「判読性 (正しく読めるか)」、「読みやすい」、「美しい」、「目が疲れにくい」、「好きな」の 5 つとする。これらは、可読性に関する印象語である⁹⁾。
- (2) 字種ごとの生成結果を基準フォントと比較し、優れている (+2)、やや優れている (+1)、同等 (0)、やや劣る (-1)、劣る (-2) の 5 段階で採点する。
- (3) 基準フォントには、人手で制作し最適化したビットマップフォント (評価対象フォントと同一画素数) を用いる。また、指標「判読性 (正しく読めるか)」に対する字体確認用として字画省略のない 48 ドットフォントも同時に提示する。
- (4) G1, G3, G4, G6 の提示順は、字種ごとにランダムに変更する。

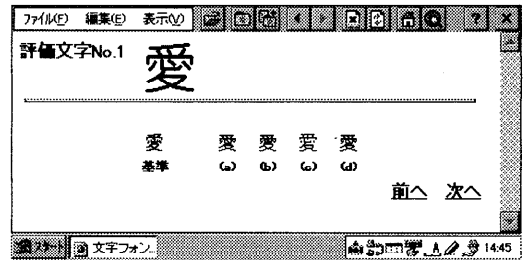


図 12 主観テストの画面例

Fig. 12 Screen shot of subjective test.

- (5) 評価対象フォントの画素数は 11, 13 および 16 ドットとする。
- (6) フォントの提示には、WindowsCE ベースのハンドヘルド PC (480×240 画素モノクロ液晶, ドットピッチ 0.25 mm/dot) とその上で動作する WWW ブラウザ (Pocket Internet Explorer) を用いる (図 12)。

実験にあたっては、単なる順位づけでなく定量的な評価を目指した。基準フォントや字体確認用フォントを提示して比較するのは、このためである。これらの提示は、つぶれて読めなくなった文字が何であるかを被験者に教えることになるが、それは評価に影響しないと考えた。

被験者から得られた各字種に対する得点を指標ごとに平均した結果を図 13 に示す。出力画素数 13 ドット時の「目が疲れにくい」という指標を除き、指標によって得点順位が入れ替わることは、ほとんどなかった。すなわち、実験に用いた指標には互いに相関があり、フォントの可読性評価に有効であるといえる。

出力画素数 11 ドット時の得点は、G1 と G6 が、ほとんど同じで最も高く、続いて G4, G3 の順となった。このとき、補間係数 $R = 0$ としているので、G6 は G1 と等価なストロークフォントデータになっている。したがって、G1 と G6 の評価が基準ビットマップフォント並となったことは、11 ドットに最適化したストロークフォントデータが、人手で制作したビットマップフォントと同等の品質を達成できたことを示している。また、G4 が G3 よりも良い評価を得ているのは、線間補正処理の有効性を示すものである。

出力画素数 13 ドットでは、基準ビットマップフォントには及ばないものの、G6 の得点が最も高く、続いて G4 がくる。この結果から、複雑度補間が有効にはたっていることが分かる。G1 と G3 の得点は、「目が疲れにくい」という指標でのみ G1 が高く、他の指標では両者変わらない。

出力画素数 16 ドット時の得点は、G6, G4, G3, G1

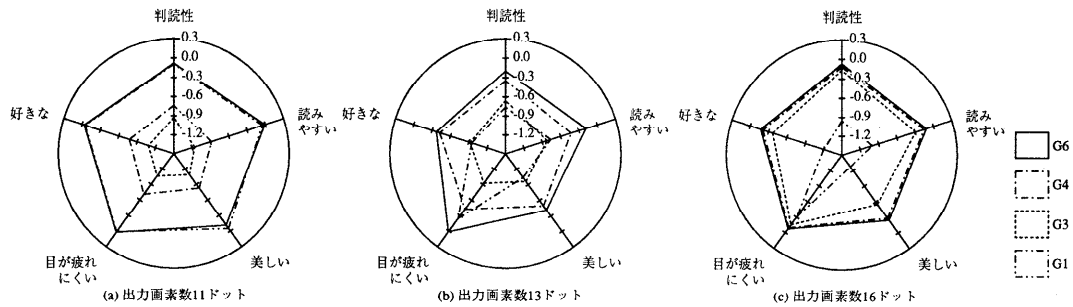


図 13 主観テストの結果

Fig. 13 Results of subjective test.

の順となった。図 9 に示すように、この画素数では形状補正を行わない G3 でもつぶれが少なく、高得点を得ている。しかし、G6、G4 は、さらに良い結果となっており、形状補正処理の効果が認められる。G1 は 11 ドット用のデザインなので、字画の省略が目立ってしまい評価が低い。しかし、「目が疲れにくい」という指標だけは 13 ドット時と同様に比較的高い得点となっている。

以上より、出力画素数が 11 ドットから 16 ドットに近付くにつれて、生成品質の順位は G1 下降、G3 上昇、G4 上昇となり、線間補正と複雑度補間を併用する G6 は安定して高い品質を得られることが分かった。また、G1 の生成結果からは、少々省略があってもすっきりとしたつぶれないフォントは、目に負担をかけないことが分かった。

5. まとめ

本論文では、文字サイズを自由に変更可能で、大きく拡大したときに正確な字体を表現できるだけでなく、小型携帯情報端末で多く用いられる低画素サイズにおいても字画につぶれない判読が容易なフォントの生成を目指して、線間補正処理と文字複雑度補間処理を用いたストロークフォント方式を提案した。被験者による主観テストによって、本論文で提案するストロークフォント方式を用いれば、字形の複雑な漢字であってもビットマップフォントに匹敵する品質で生成できることを確認した。また、本方式のデータサイズはパソコンなどで使用されているアウトラインフォント方式よりも小さくなり、大容量記憶装置を搭載できない携帯端末でも使用可能な見通しを得た。

本フォント方式は、今後ますます普及するであろう携帯情報端末のユーザインタフェースの改善に寄与するものと考えられる。

謝辞 日頃ご指導いただく三洋電機(株)ハイパーメディア研究所鈴木治所長に感謝します。また、フォ

ントデータを制作していただいた同通信システム研究部ヒューマンインタフェース研究室高木多加予さんに感謝します。

参考文献

- 1) 日本工業標準調査会：人間工学—視覚表示装置を用いるオフィス作業—視覚表示装置の要求事項 JIS Z 8513, 日本規格協会 (1994).
- 2) Knuth, D.E.: *The METAFONTbook*, Addison Wesley (1986).
- 3) 本門慎一郎, 西村幹男, 平野文昭：パーツセパレート方式の漢字アウトラインフォント技術, *FUJITSU*, Vol.46, No.6, pp.598-604 (1995).
- 4) 山崎一生, 入山徳夫：高精細デジタル文字図形の変倍の一方法, *電子情報通信学会論文誌 (D)*, Vol.J70-D, No.2, pp.387-396 (1987).
- 5) 安本 護, 池田尚志, 中里西克, 富田義数：モバイル環境のユーザインタフェース向上に向けて—携帯情報端末用フォントの高品質化に関する一手法, *インタラクシオン'98 論文集*, pp.49-56, 情報処理学会 (1998).
- 6) 日本規格協会文字フォント開発・普及センター：平成 5 年度マルチメディア対応フォント調査研究報告書 (1994).
- 7) 陳 和明, 小沢慎治：多様な明朝体文字の規則的な生成, *電子情報通信学会論文誌 (D-II)*, Vol.J72-D-II, No.9, pp.1423-1431 (1989).
- 8) 田中哲朗, 岩崎英哉, 長橋賢児, 和田英一：部品合成による漢字スケルトンフォントの作成, *情報処理学会論文誌*, Vol.36, No.9, pp.2122-2131 (1995).
- 9) 宮崎紀郎：新聞紙面における可読性—読みやすさへの実験を通して, *印刷雑誌*, Vol.76, No.12, pp.17-23 (1993).

(平成 10 年 5 月 18 日受付)

(平成 10 年 12 月 7 日採録)

**安本 護 (正会員)**

1959年生。1982年神戸大学工学部電子工学科卒業。1984年同大学大学院工学研究科電気工学専攻修士課程修了。1998年岐阜大学大学院工学研究科電子情報システム工学専攻博士課程修了。博士(工学)。1984年三洋電機(株)入社。フォント処理技術に関する研究に従事。電子情報通信学会会員。

**池田 尚志 (正会員)**

1944年生。1968年東京大学教養学部基礎科学科卒業。同年工業技術院電子技術総合研究所入所。制御部情報制御研究室、知能情報部自然言語研究室に所属。1991年岐阜大学工学部教授。工学博士。自然言語処理の研究に従事。人工知能学会、電子情報通信学会、言語処理学会、ACL各会員。

**中里 西克**

1968年生。1991年名古屋大学工学部応用化学科卒業。同年三洋電機(株)入社。フォント処理技術に関する研究に従事。電子情報通信学会会員。

**富田 義数**

1951年生。1973年大阪大学工学部通信工学科卒業。同年三洋電機(株)入社。以来、主に放送関連の研究開発に従事。1996年より、ハイパーメディア研究所にてヒューマンインタフェース、移動体通信関連の開発を担当。現在、同研究所通信システム研究部部长。映像情報メディア学会、電子情報通信学会各会員。