漢字画像データにおける横線と縦線の字画の抽出に基づく漢字フォントの自動生成†

奥村彰二†† 佐藤義雄††

漢字フォントを製作するには，レタリングされた漢字を画像データとして計算機に入力し，出力結果に対する人の判断を入れながら，対話的に処理を行うのが非常に有効で確実な方法と考えられる．しかし，漢字のいろいろな種類や大きさに対応しようとすると，漢字の数が多いために，対話的処理の際の人の労力の負担が大きい．この論文では，ファイルとして蓄積された漢字画像から，任意サイズの漢字フォントを計算機により自動生成する一つの方法を示す．明朝体の漢字画像における横線および縦線の字画を抽出し，それらの始点と終点の位置および他の字画との接続状態を示すコードを保存する．その他の主に曲線を輪郭線として持つ字画については，チェイン符号を利用してそれらの輪郭線を保存する．漢字フォントのドットパターンは，これら二種類のデータより発生したドットの組み合わせとして発生する．この方法により任意サイズの十分満足できる漢字フォントが自動生成されることを示す．

1. まえがき

近年，レーザービームプリンタを主とするページプリンタが飛躍的に高性能化，低価格化するのに従い[1]，これらを用いて日本語文章を出力する際に必要となるデジタルの漢字フォントに対して，いろいろなサイズや字体の変化の要求が増大してきている[2),3)]．計算機出力用の漢字フォントは，我が国において既に数多く作られているが，字体には流行や個人的な好みがあること，そのシステム独自の字体が求められる場合があることなどの理由で，今後も新たな漢字フォントが作られていくものと思われる．

漢字フォントを作る方法として，計算機への漢字の形状を表すデータの入力，漢字フォントの発生と表示および修正等の過程を繰り返す CAD の試みがこれまでにいくつか報告されている[4)~8)]．これらの方法においては，漢字フォントの一つのセットで取り扱う漢字の数が7000以上と多いため，計算機との対話的処理における人間の負担が非常に大きい．一方，デザインされた漢字をビデオカメラなどを用いてデジタルな画像データに変換し，これらの計算機処理により漢字フォントを発生することも通常行われている．この処理においても，入力された画像データから一次的な漢字フォントを発生した後は，漢字フォントの表示と修正の過程を繰り返すことになり，計算機の対話的使用における人間の労力は，やはり相当大きなものとなっている．この論文では，漢字画像データを計算機により処理し，その間，人間による援助なしに，任意のサイズの漢字フォントを自動的に生成する一つの方法について論じる．

明朝体のような幾何学的な字体において，同じサイズの漢字フォント，または一つの漢字の中で，横線や縦線の字画の線幅や形状が揃っていることが，デジタルの漢字フォントでは好ましい条件と考えられる．計算機に取り込まれた漢字画像データを，そのままフォントサイズに合わせて縮小（または拡大）しようとする方法では，この条件を満足させることが困難である．一方，'てん'，'はらい' や 'はね' 等，輪郭が曲線である字画の部分はもともと形状に変化があり，漢字の間で多少の不揃いがあっても，輪郭線が滑らかであれば，文字の美しさが特に損なわれることはない．これらのことを考慮して，我々は明朝体の漢字の画像データから，横線と縦線の字画の位置と長さを検出して，それらに対応するドットパターンを一定の規則で発生する．この際，'うろこ' と呼ばれる横線の右端の三角形，縦線の 'あたま' や 'まがり' における膨らみなど，明朝体特有の修飾の形状を一定の方法で発生させる．曲線の輪郭を持つ字画の部分は，画像パターンをそのままフォントサイズに応じて縮小（または拡大）することによりドットを発生する．このような異なった二つの方法で発生したドットを組み合わせて自動生成された漢字フォントパターンの例を図1に示す．この図で，# と＊はそれぞれ，横線と縦線として発生したドットを示し，& は画像データの縮小で発生した字画の要素のドットを示す．以下の章で，この方法の手順を述べ，これにより十分満足な任意サイズの

† Automatic Kanji Font Generation Based on Extraction of Horizontal and Vertical Line Strokes in Kanji Image Data by SHOJI OKUMURA and YOSHIO SATO (Department of Information Science, Faculty of Engineering, Fukui University).

†† 福井大学工学部情報工学科

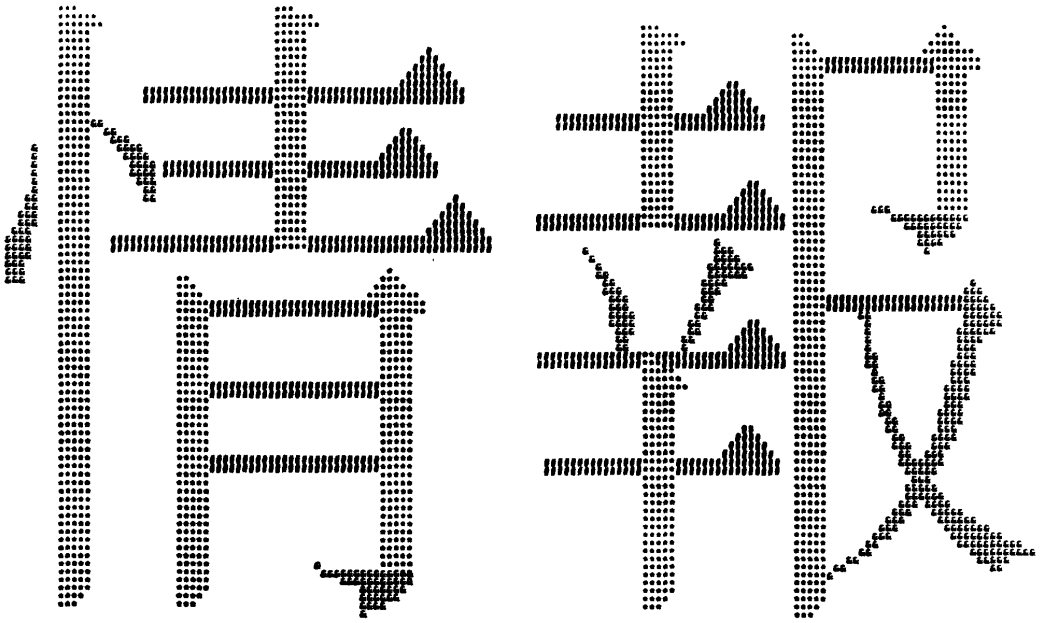


図 1 計算機により生成された漢字フォントドットパターン
Fig. 1 Kanji font dot patterns generated by a computer.

漢字フォントが自動生成されることを示す.

2. 漢字画像データから横線字画の抽出

使用した漢字画像データは，レタリングされた明朝体文字をビデオカメラでデジタル入力された 384×384 の大きさの2値画像である．まず前処理として，漢字画像データから雑音を取り除く．そのために，全画素について，一つの画素とその8隣接画素から成る3×3 の領域に存在するドットに，**図2**(a)で示される重みをつけて計算される和 S について，$S \leqq 13$ のとき，真ん中のドットを消去し，$S \geqq 14$ のとき，真ん中の画素にドットを置く．横線や縦線の輪郭には，1ドットの凹凸の雑音として，図2の(b)および(c)，またはそれらを 90° ずつ回転して得られるパターンがしばしば現れる．この操作により，それらは適当に修正される．

明朝体漢字の横線字画の抽出では，右端に付く‘うろこ’も含めて，その輪郭の下側の線が水平な線分になっていることに着目する．一つの漢字画像データについて，縦の画素の各列について上から下へ走査して，縦方向のドットの連なり（ラン）の始点と終点の位置を保存する．ただし，このランレングスが，あらかじめ予想される横線字画の太さの範囲から外れているものは無視する．ここで，画素の位置を定義するために，2値画像の左上隅を原点とし，そこから下の方に y 座標を，また右の方へ x 座標を設定する．このように得られたデータをランデータと呼ぶことにする．一つの漢字の画像について，この終点の y 座標の分布を，ヒストグラムとして表したものを**図3**に示す．これらの頻度分布に一定のしきい値を設定することにより，横線字画の存在とその y 座標が推定される．このしきい値は，数個の漢字を処理して経験的に決めているが，明らかに，値を高く設定すると横線を見落とし，低く設定すると偽の横線を抽出する場合が生ずる．次に，ランデータから終点の y 座標が，推定された横線の水平位置（y 座標）

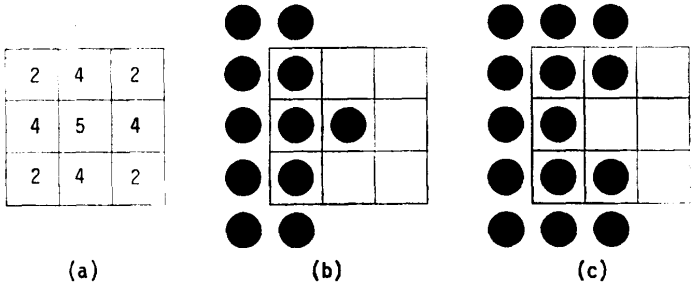


図 2 (a) 画像の平滑化操作における隣接 3×3 画素の重み値
(b), (c) 字画の輪郭部における1ドットのノイズの例
Fig. 2 (a) Weights of 3×3 neighbor pixels for image smoothing.
(b), (c) Examples of 1-dot noise at the edge of a stroke.

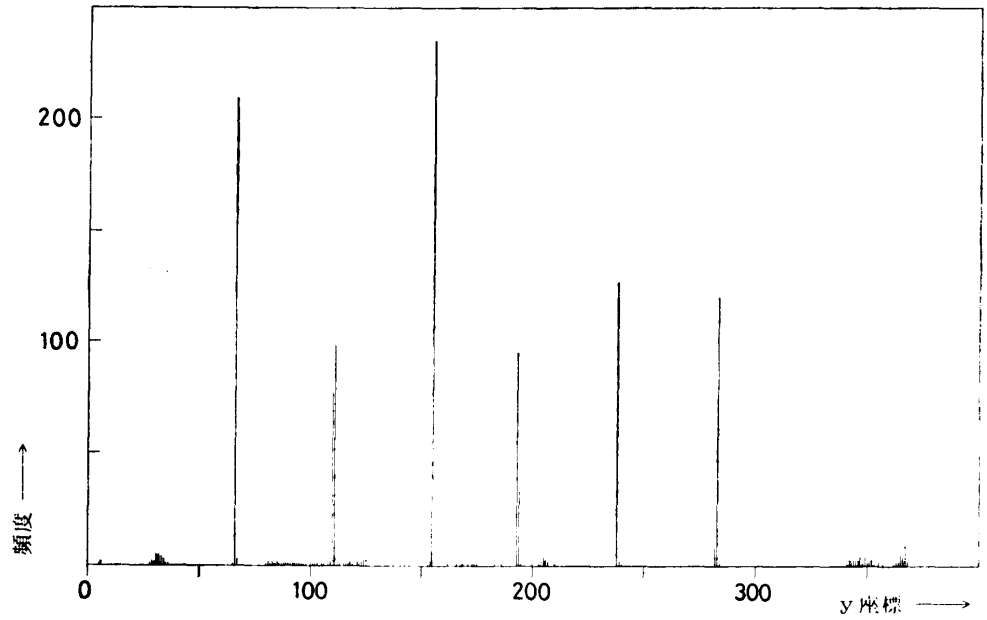


図 3 漢字2値画像における垂直方向のドットのランの終点の y 座標位置分布を表すヒストグラム (漢字 '情' に対して)

Fig. 3 A histogram for the y-coordinates of the end points of vertical runs of dots in the binary Kanji image (情).

との差が一定値以下のものを選び出す．この選び出されたデータが横線によるものであれば，それらの x 座標は，連続した一連の整数値になるので，これらの最小値と最大値から横線の始点と終点の位置が検出できる．横線に他の字画が交叉していると，そこでのランレングスが本来の横線のそれより大きくなり，横線上のドットのランがデータから抜けることになる．字画の意味では一本の横線が，この場合には，複数の横線として検出される．そのために，もとの画像データにおいて，同じ y 座標の二本の横線の間にドットが連続的に存在しているかどうかを調べて，それらはもともと，1本の横線であるか否かを判断する．また，一つの横線と見られるドットパターンについて，さらに輪郭の下側の線の水平な直線に対する傾斜が一定値以下であることを確認する．横線の検出に画像データでの輪郭線の水平度に着目しているので，漢字画像を傾けて入力されると，データの前処理で正しい方向に修正されない限りうまくいかない．現時点では，画像は正しい方向で入力されていると仮定し，そのような前処理は行っていない．最後に 'うろこ' の部分も含めて，抽出された横線に対応する画素のドットを画像データから消去する．

3. 漢字画像データから縦線字画の抽出

縦線の抽出には，画像データを 90 度回転させて考えるだけで，横線の場合とほぼ同じ手順で行われる．明朝体漢字の縦線の 'あたま' には，右の方に膨らみがあるので，縦線の輪郭線のもつ左側の垂直な線分に着目する．既に抽出された横線のドット列が除かれた画像について，水平方向に画素を走査して水平方向のドットの連なり（ラン）を探し，その始点と終点の位置を集めたランデータを作る．ただし，このランレングスが，あらかじめ予想される縦線字画の太さの範囲から外れているものは無視する．ランデータの中で始点の x 座標に着目して，縦線字画の始点と終点の位置を検出する手順は，横線の場合と全く同じである．検出された縦線について，垂直な直線に対する傾斜を調べて，それが縦線字画であることを確認する．また抽出された縦線に対応するドットは，画像データから消去する．

4. 漢字フォントデータファイルの形成

前章で述べた方法により抽出された結果を，フォントデータとして一つのファイルに出力する．横線字画については始点と終点の x 座標と，輪郭の下側の直線の位置を表す y 座標を保守する．また縦線について

は，始点と終点の y 座標と，輪郭の左側の直線の位置を表す x 座標を保存する．さらに，横線または縦線の始点および終点において，他の字画または字画の要素との接続状態を示す1バイトのコードを付け加える．図1の例で示しているように，抽出された縦線には，'はね' や 'はらい' などの字画の要素と続きになっているものも含まれている．この状態コードは，漢字フォントのドットパターンを発生させる際に，横線と縦線の端の形状の決定に使用される．例えば，横線の終点が開放状態（他の字画との接続がない）であれば，そこに 'うろこ' を付けることにする．この情報を得るために，横線と縦線の始点と終点の近傍における画像データのドット分布を調べる．また，横線と縦線の相互位置関係を得るために，既に抽出された横線と縦線の位置データも利用する．ファイルに保存する一つの横線に対する位置データと状態コードの形式を図4に示す．縦線に対しても，同じような形式を採用している．横線と縦線が除かれた画像データについては，

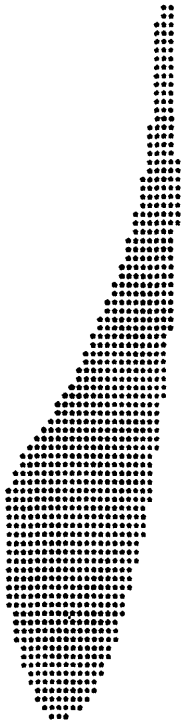
縦線のデータ形式

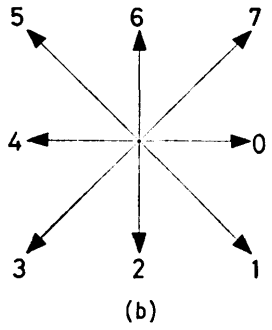
縦線の位置（x 座標）	(9 bits)
始点の位置（y 座標）	(9 bits)
終点の位置（y 座標）	(9 bits)
始点における状態コード	(1 byte)
終点における状態コード	(1 byte)

縦線の始点における状態コード (1 byte)

状態コード	意　味
0	開放状態（他の字画との接続なし）
2	横線以外の字画に接続
$10+X$	X 番目の横線の始点に接続
$40+X$	X 番目の横線の中間点に接続
$70+X$	X 番目の横線の終点に接続（まがり）

図4 漢字フォントデータにおける縦線のデータ形式
Fig. 4 Data format of a vertical line stroke in the Kanji font data.





(b)

(0, 1) (2,17) (1, 1) (2, 7) (3, 1) (2,11) (3, 1) (2, 7) (3, 1) (2, 5)
(3, 1) (2, 5) (3, 1) (2, 3) (3, 1) (2, 2) (3, 1) (2, 2) (3, 1) (2, 2)
(3, 1) (2, 2) (3, 1) (2, 1) (3, 1) (2, 1) (3, 1) (2, 1) (3, 2) (4, 1)
(3, 1) (4, 2) (5, 2) (6, 1) (5, 1) (6, 1) (5, 1) (6, 2) (5, 1) (6, 3)
(5, 1) (6,13) (7, 1) (6, 1) (7, 1) (6, 1) (7, 5) (6, 1) (7, 3) (6, 1)
(7, 1) (6, 1) (7, 1) (6, 1) (7, 1) (6, 1) (7, 1) (6, 1) (7, 1) (6, 1)
(7, 1) (6, 1) (7, 1) (6, 2) (7, 1) (6, 2) (7, 1) (6, 3) (7, 1) (6, 5)
(7, 1) (6,11) (7, 1) (6, 1)

(c)

(0,2) (1,17)
(3,2) (1,11)
(3,2) (1,11,1,7,1,5,1,5,1,3,1,2,1,2,1,2,1,2,1,1,1,1,1,1,2)
(4,3) (1,1,2)
(5,6) (2,1,1,1,1,2,1,3,1,13)
(7,6) (1,1,1,1,5,1,3,1,1,1,1,1,1,1,1,1,1,1,1,1,1,2,1,2,1,3,1,5,1,11,1,1)

(a)　　(d)

図5 (a) 画像ドットパターンの一例（情の字の左側のてん），(b) チェインコードの割当，(c) 形式（チェインコード，繰り返しの数）による(a)の輪郭線の符号化，(d) 形式（チェインコード1，チェインコード2）（二つのチェインコードの交互の繰り返し数の列）で表した(c)と等価な表現
Fig. 5 (a) An example of an image dot pattern. (b) An assignment of a chain code. (c) The contour coding by means of the chain code as a format of (chain code, number of repetitions). (d) The equivalent expression of (c) as a format of (chain code 1, chain code 2) followed by (sequence of their alternate repetitions).

表 1 漢字フォントデータの大きさ（バイト）
Table 1 Sizes of Kanji font data (byte).

漢字	データ量（バイト）	漢字	データ量（バイト）	漢字	データ量（バイト）	漢字	データ量（バイト）
亜	52	茜	212	圧	112	或	542
唖	76	穐	455	斡	408	粟	411
娃	357	悪	321	扱	688	袷	478
阿	277	握	445	宛	556	安	580
哀	403	渥	546	姐	361	庵	456
愛	887	旭	262	虻	237	按	707
挨	634	葦	151	飴	503	暗	203
姶	524	芦	135	絢	578	案	648
逢	535	鯵	969	綾	882	闇	285
葵	663	梓	310	鮎	394	鞍	542

その輪郭線をフリーマンのチェインコード[9]として保存する．使用されたチェインコードは，**図5**(b)のように設定し，3ビットで表される．チェインコードは，同じものが連続して現れる傾向があるので，チェインコードとその繰り返しの数というデータの組で扱う．例として，図5(a)に示された画像ドットパターン（'情'の字の立心偏における左の点）に対する左上から始まり，右周りの輪郭線のチェインコードを図5(c)に示している．ここで，括弧の中の左の数がチェインコードで，右の数はその繰り返しの数である．この例からも明らかなように，漢字の輪郭線におけるこのような扱いでのチェインコードは，連続する二つの整数の対の繰り返しとなる確率が高い．したがって，図5(d)に示すような二つのチェインコードとそれらの交互の繰り返しの数という組み合わせを，必要とするビット数で表して保存した．以上の方法で表したJIS第一水準漢字の最初の40字に対するフォントデータの大きさを**表1**に示す．表1で明らかなように，横線と縦線のみで形づくる'亜'や'唖'のような漢字では，特に少ないデータで表現される．

5. 漢字フォントドットパターンの生成

前章までの漢字画像データから漢字フォントデータの生成とは独立な計算機処理により，一つの漢字フォントデータを入力として，漢字フォントのドットパターンを自動生成する．横線と縦線については，もとの画像データの精度（384×384）で位置と長さが与えられるので，それらをフォントサイズに比例して縮小（拡大）して，対応するドットパターンを一定の方法で発生する．すなわち，始点および終点に対しての条件コードを参照しながら，横線に対しては，一定の幅の横のドット列と'うろこ'，縦線では，一定の幅の縦のドット列，'あたま'や'まがり'などを発生する．ここでは，単なる矩形や三角形の領域にドットを満たすだけで，その大きさを自由に変えることは容易である．一方，輪郭線を与えるチェインコードから，横線と縦線が除かれた二値画像データを復号する．画像領域を縦横それぞれフォントサイズに等しい数のメッシュで分割する．一つの矩形メッシュの中での画像データのドットの分布より，漢字フォントのその位置での1ドットの有無を決める．この決め方はいろいろ考えられるが，我々は，一つのメッシュ内のドットの分布を，その真ん中の3×3の画素で代表させて，そこでのドットの数が5以上のときドットを落とし，4以下のとき空白とした．

この方法では，横線や縦線の太さ，すなわちドット幅は，漢字フォントの大きさに比例させる必要はなく，文字として適当であれば，任意の寸法に設定できると考えられる．横線については，かなり自由に設定可能である．ただし，縦線については，前述のように，'はね'や'はらい'などの字画の要素に続く場合に，つなぎの部分で滑らかに接続されなければならない．輪郭線で与えられた漢字の字画の太さを変えるために，字画の中心線を定義し，そこから輪郭線までの距離を一様に伸び縮みさせる方法が考えられる．字画の方向が決め難い，'はね'や'てん'のような短い字画の要素では，画像データから中心線を自動的に見つけることが非常に困難である．ここでは，輪郭線データから得られるドットパターンを，水平方向にのみ，部分的に拡大または縮小して縦線に接続している．しかし，この方法で太さを変える度合いを増していくと，輪郭線から発生する字画の歪みが目立ってくるので，今のところ，縦線の太さを自由に変化させることができない．一つの漢字フォントデータから生成されるフォントのサイズは1ドットのステップで任意サイズに指定でき，また，いろいろなサイズのものを同時に生成することも可能である．

6. 実験例と検討

自動的に得られた一つの漢字フォントデータから，生成された漢字フォントの例を**図6**に示す．図6では，フォントサイズが40×40から100×100まで，縦横4ドットずつ変えるように計算し，解像度が300 dots/inchのプリンタで出力している．同じ大きさの

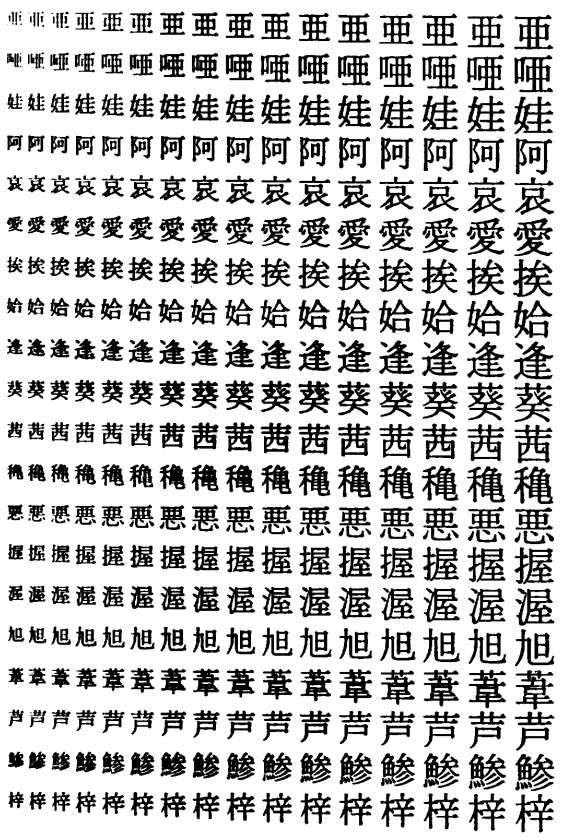


図 6 漢字フォントの出力例
Fig. 6 Samples of generated Kanji fonts.

フォントの中では，その方法から明らかなように，縦線や横線に線幅のばらつきは生じていない．もともとの横線と縦線が，他の字画との関係で抽出されないことがあれば，その部分は輪郭線としてフォントデータに取り込まれる．この時は，画像データの形状がそのまま現れたり，フォントデータが大きくなることなど好ましくない結果が生じる．このようなことの起こる漢字は，JIS 第一水準の漢字についての予備実験[10]では，約50文字ぐらいあったが，検出のアルゴリズムの改善で今後さらに少なくできると思われる．

この研究では，CPU として MC 68020 (16 MHz) を持つ計算機システムを使用した．プログラムは，すべてCで書かれている．計算時間は，漢字画像より漢字フォントデータを生成するのに1文字当り約5秒，フォントデータから一つの漢字フォントのドットパターンを生成するのに，64×64 のサイズで約3秒である．これらの計算時間にはディスクの読み書きの時間も含まれている．漢字画像データを磁気テープやディスクにあらかじめ用意しておけば，計算機により漢字フォントが自動的に生成されるので，この程度の計算時間は，現実にはあまり問題にならないと考えられる．

7. むすび

漢字画像データの中で横線と縦線の字画を抽出する方法で，十分満足のいく任意サイズの明朝体の漢字フォントが自動生成できた．横線と縦線を除いた画像データの符号化および復号法の過程に，適当な範囲の誤差を認めて，漢字フォントデータをより小さくすることが望まれる．他の字体についての画像データからの自動生成も今後の課題である．ゴシック体については，明朝体より横線と縦線の抽出が容易であるので，ここでの方法が少々の修正を加えれば適用できると予想される．ほとんどの字画の輪郭が傾きのある直線または曲線で構成される教科書体などの字体については，横線と縦線の抽出にかなりの難しさが予想され，輪郭線の自動的で有効な符号化方法を検討しなければならない．

謝辞 本研究の遂行にあたり，いろいろと御援助くださいました日立工機(株) 徳永一美氏，日立製作所(株)デザイン研究所 日向秀予氏および中村直喜氏に感謝いたします．また，この研究について御議論いただいた同デザイン研究所の中田光氏，坂本達氏ほかの方々に感謝いたします．

参考文献

1) 手嶋，根本，中沢：プリンタ新時代，日経バイト，1987年12月号，pp. 125-167.
2) 住永：日本語デスクトップ・パプリッシング，日経エレクトロニクス，No. 435, pp. 127-141 (1987).
3) 大用：ディスクトップ・パプリシングの最前線を探る，日経バイト，1987年12月号，pp. 110-123.
4) 山崎，井村：文字輪郭線の円弧と直線とによる近似，情報処理学会論文誌，Vol. 26, No. 4, pp. 726-732 (1985).
5) 坂元，高木：高品質明朝体ひらがな・カタカナフォントの計算機による生成，電子通信学会論文誌，Vol. J 68-D, No. 4, pp. 702-709 (1985).
6) 張，真田，手塚：漢字楷書毛筆字体の計算機による生成，電子通信学会論文誌，Vol. J 67-D, No. 5, pp. 559-606 (1984).
7) 戸倉，鈴木，中村，牧野，高倉：つづけ字を可能とする毛筆体文字生成システム，情報処理学会論文誌，Vol. 29, No. 1, pp. 20-28 (1988).
8) 曾，井上，真田，手塚：毛筆文字デザインエキスパートシステムのための書道知識の解析とルール化の考察，情報処理学会論文誌，Vol. 29, No.

2, pp. 160-168 (1988).
9) Freeman, H.: Computer Processing of Line-drawing Images, *Comput. Surv.*, Vol. 6, pp. 57-97 (1974).
10) 奥村，宇野，佐藤：漢字画像データから漢字フォントの自動生成，第32回情報処理学会全国大会論文集，pp. 1509-1510 (1986).

(昭和63年 3 月 9 日受付)
(昭和63年10月 7 日採録)

奥村 彰二（正会員）

昭和14年生．昭和37年京都大学工学部原子核工学科卒業．昭和39年同大学院修士課程修了．昭和41年同大学院博士課程中退．同年東京大学原子核研究所助手．昭和46年高エネルギー物理学研究所助手．昭和47年同助教授．昭和51年福井大学工学部助教授．昭和53年同教授．理学博士．現在は，コンピュータグラフィックス，図形・文字入出力システム，画像処理，マイクロプロセッサ応用システムなどの研究に従事．日本物理学会，電子情報通信学会各会員．

佐藤 義雄（正会員）

昭和23年生．昭和46年名古屋大学工学部応用物理学科卒業．昭和50年同大学院工学研究科情報工学専攻修士課程修了．昭和53年同博士課程修了．現在福井大学工学部情報工学科助手．数値解析，図形処理，パターン認識などに興味を持つ．