漢字画像から文字要素の自動抽出†

奥村彰二†† 前田正弘††*

計算機内に漢字の文字フォントを保持する方法として，漢字の字画に近い文字要素をデータの基本要素とし，これらの組み合わせで漢字を構成する要素組立法がある．この論文では，明朝体の漢字を対象とし，漢字全体に共通のあらかじめ設定した文字要素を漢字画像から抽出し，漢字の再構成に必要なデータを計算機により自動的に作成する方法について論じる．まず漢字画像に細線化を施すことにより，線図形としての構造と太さの情報を得る．それらを端点，交点，角を節とし，枝にはこれらの節を結ぶ線分（セグメント）を割り当てたグラフ構造を作成する．計算機内のグラフ構造における節や枝に，幾何学的な寸法，位置，形状などの情報を与える．次に，グラフ構造をたどって，隣り合うセグメントの形状や太さの変化を調べていくことにより，文字要素を構成する複数のセグメントの組み合わせを見出す．最終的にその組み合わされたものを文字要素とし，その形状の種類を決定し，再構成の時に必要な幾何学的なデータを取り込む．フォントサイズ 384×384 の明朝体の第一水準の漢字フォント 2965 個に適用した結果，認識率は 77% であった．細線化，セグメントの統合，文字要素の識別，データの収集における問題点，およびその解決方法などについて論じられる．

1. はじめに

最近，日本語の文書作成を計算機によって行う機会が多くなっているが，いろいろな書体と大きさの高品位の文字フォントを用いて文書を出力したいという要求は，現在でも十分には満たされてはいない．漢字フォントのパターンをデジタル化して得られるラスタデータを，そのまま計算機内に保持する方法では，データ量が膨大となること，文字の拡大や縮小が柔軟に処理できないことなどの欠点があるため，それに代わる計算機内部の文字の表現法がいろいろと提案されてきている．漢字の輪郭線を符号化して保持する方法（アウトライン方式）は，この欠点をかなり軽減しており，現在広く利用されているが，それでも，ラスタデータに戻す時の太さのばらつきの発生，文字の形状や太さが少し変化するごとに，それに対応してデータを用意しなければならない等の問題を残している．

これに対して各漢字ごとに，独立にその形状を表現するのではなく，あらかじめ漢字を構成する文字要素（縦線，横線，点など）を設定し，それらを組み合わせて文字を再構成する方法[1)~4)]が提案されている．この方法は要素組立法またはエレメント方式と呼ばれているが，漢字フォントは，数十の文字要素の中からの選択と，それらの大きさおよび位置を示すデータで表される．漢字の出力においては，これらのデータに基づいて各々の文字要素を描画する共通のルーチンを用意すればよいので，文字フォント全体のデータ量を著しく縮小することが可能である[3),4)]．また，漢字フォントを新たに作成するという観点から考えると，個々の文字要素を生成する方法における変更により，同じフォントデータからある程度異なった書体の漢字を生成する可能性もある[2)]．

このような漢字フォントのデータを得るために，最初からこの方法によって新たな漢字フォントをデザインしていくことも考えられるが，それは美的感覚を伴う専門的な仕事であり，データの処理のみではない多くの解決すべき問題が関わってくる．より手間の少ない機械的な処理法として，既に存在する漢字の画像データから文字要素を識別して，その位置や大きさの情報を抽出・収集して漢字フォントを構成する方法が考えられる．このような作業を計算機を用いて行う最も一般的で，確実な方法の1つは，ディスプレイ端末に漢字画像や処理の中間結果を表示し，人間による確認や選択を対話的に取り入れながら，漢字画像データを処理していくことである．しかし，試行錯誤が加わったり，違った種類の書体をも対処しようとすると，この方法でも，漢字の数が多いために作業者の負担が非常に大きい．そのために要素組立法のデータの作成を，計算機によってなるべく自動的に行うことが特に望ましいと考えられる．これまで文字認識の研究は数多く行われてきたが，文字画像における字画または文字要素そのものの識別，およびそれらの正確な形状の抽出を目的にした研究は，今までほとんど行われていない．

本論文では，明朝体の漢字画像データから要素組立法による漢字フォントデータを自動生成するために，

† Automatic Extraction of Character Elements in Images of Chinese Characters by SHOJI OKUMURA and MASAHIRO MAEDA (Department of Information Science, Faculty of Engineering, Fukui University).
†† 福井大学工学部情報工学科
* 現在 オージー情報システム総研(株)

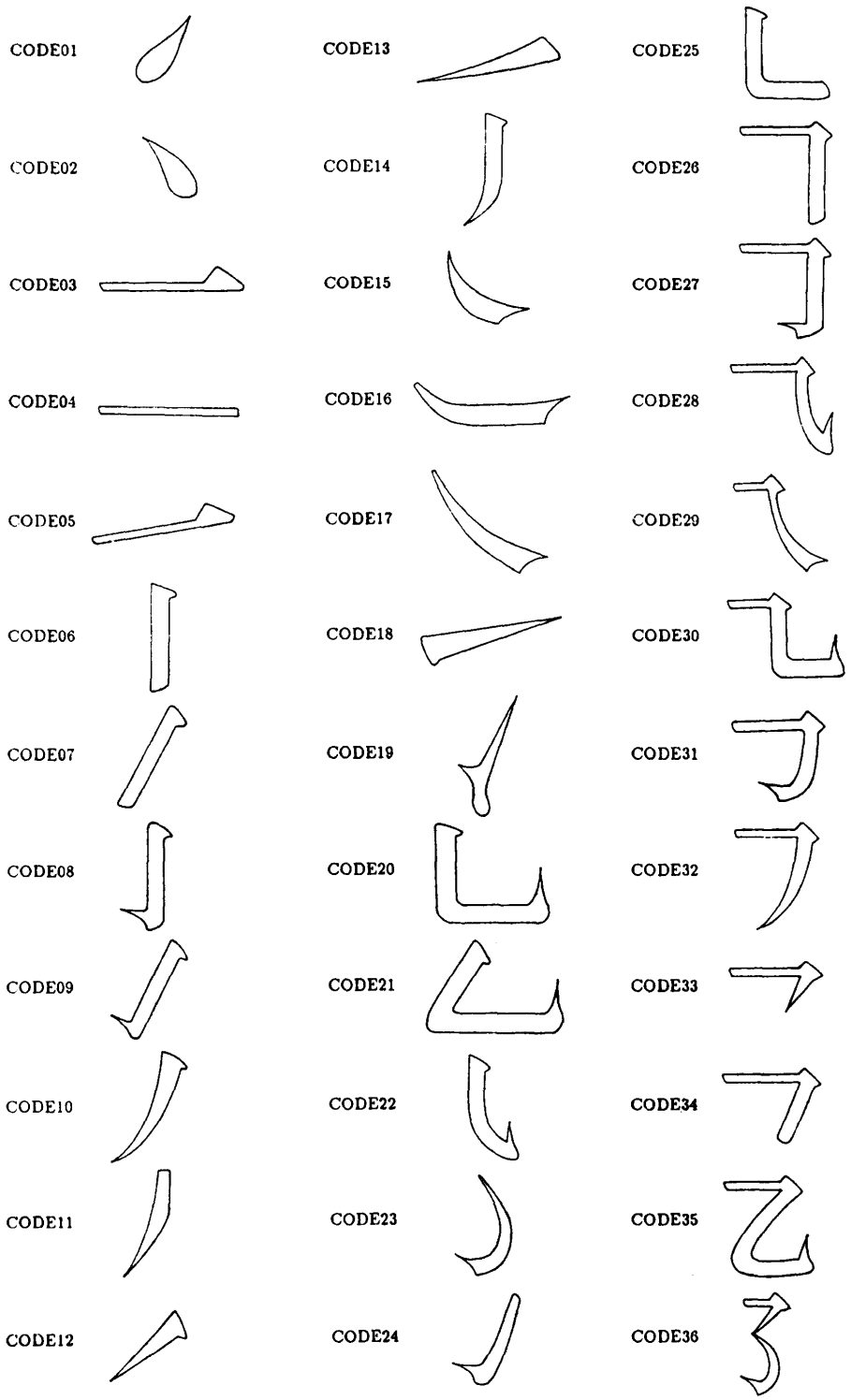


図 1 設定した漢字を構成する文字要素
Fig. 1 Selected character elements constructing the chinese characters.

設定した文字要素の種類と形状情報，それらの抽出方法および実行結果などについて論じられる．

2. 抽出すべき文字要素データと処理の流れ

前章で述べたように要素組立法では，1つの漢字をあらかじめ設定した文字要素に分解するが，それらは文字要素の描画ルーチンにおいて，直線や曲線の並行移動，回転，全体または部分的な拡大および縮小などの一定の処理によって生成可能なものとする．例えば，水平な横線は漢字の中で頻繁に現れるが，それが横線であることと，始点と終点の位置および太さが与えられれば，もとの形状を描くことができる．明朝体における横線の右端に付いている「うろこ」と呼ばれている三角形に似た形状も，横線の長さの変化に対応した一定のルールで生成可能であるので，1つ1つの横線の輪郭を保存する必要はない．縦線や「てん」などについても同じようなことがいえる．次々と文字要素の描画ルーチンを働かせて決められた大きさと位置に1つの漢字を描くことを，本論文で漢字の「再構成」と呼ぶことにする．各々の文字要素の描画ルーチンの手法は，この要素組立法の成否を決める重要な部分であるが，本論文では，このことについては特に議論しない．

漢字を再構成するために，個々の漢字に対して保存するデータとしては，なるべく冗長性の少なく，形状を十分に定義するものでなければならないが，画像データから抽出すべき情報として次のものを考える．

(**文字要素の種類**) 個々の文字要素に対する描画ルーチンにおいて，位置と大きさの指定によって生成可能であるという条件の下に，漢字の字画そのもの，またはそれを適当に分解したものを選び，**図1**に示すような36種類を設定する．

(**位置および大きさ**) 文字要素の位置として，それらの中心線上で，始点（起筆点），終点（終筆点），角，「はね」の開始点の座標．これらの点を，本論文では「代表点」と呼ぶ．形状は文字要素の種類で決まるものとするので，始点と終点が与えられれば，その2次元的な大きさも自動的に決まる．

(**太さ**) 一様な太さの文字要素では，その太さの値．太さが一様でないものに対しては，最も太いところでの太さ．角を含み，そこで複数の部分に分けられる文字要素に対しては，各部分の太さ（例えばCODE 26では横線と縦線の太さ）．

(**他の文字要素との接続の有無**) 始点と終点における他の文字要素との接続の有無．漢字を再構成する時，このことの違いによって，始点と終点での形状の整え方が異なる．例えば，縦線の上端において横線と接触していれば，「おさえ」接触がない場合に比べてやや小さめに，またその上にはみ出ないように生成しなければならない．これは，「十」と「丁」における縦線の始点の描き方を比べれば明らかである．

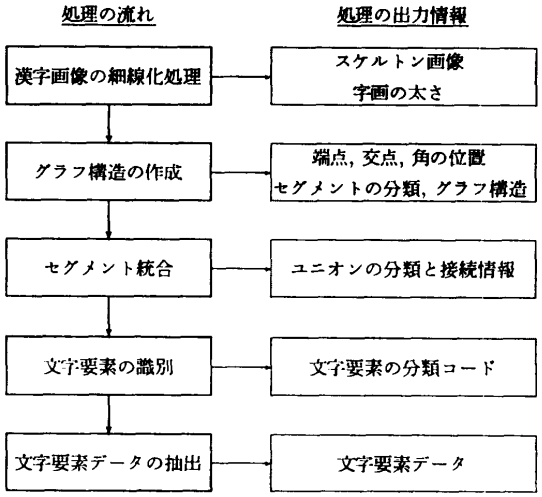


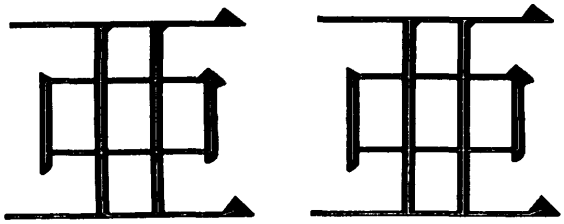
図2 文字要素の抽出に対する処理の流れとそれぞれの処理段階での出力情報

Fig. 2 A flow of processing for the character element extraction and output information of each processing step.

本論文の対象としている全体の処理の流れと，各処理の段階で得られる情報を**図2**に示す．各処理段階における入力となるデータは，主に1つ前の処理で得られたものであるが，それ以前に得られたデータを参照することもある．以下，この処理の順序でそれらの内容について詳述する．

3. 細線化処理

細線化処理[5)~8)]は，幅を持つ図形の2値画像から，その中心線（スケルトン）を取り出す処理であり，元の図形のトポロジカルな性質が保存されるので，いろいろな2値画像処理の前処理としてしばしば用いられる．ここでも，漢字の線図形としての構造を明らかにし，字画の始点や終点および交点の位置の検出を容易にするために，これを漢字画像に適用する．一般的な細線化処理では，図形の周辺部を上下左右から1画素ずつ一様に削り取っていき，黒画素の連続性（4連結または8連結[7)]）を失うところでその削除を控えて，



(a) (b)

図 3 細線化処理の結果の例
(a) 通常の細線化処理, (b) 字画の交差部で内挿処理を取り入れた細線化処理
Fig. 3 Examples of the thinning operations.
(a) Conventional thinning operation, (b) Thinning operation using the line interpolation at the intersection of the strokes.

1画素の幅で黒画素の連続性を保つようにする．このような細線化処理を漢字画像に一様に施すだけでは，後の文字要素の抽出に適した線図形が必ず得られるとは限らない[8]．**図 3-(a)** の例で示されるように，縦および横線の字画がT字形状に接続した部分においては，横線のスケルトンに，字画を示す線としては好ましくない歪みが生じる．これは，細線化処理の過程で，より細い横線の部分が先にスケルトンに達し，その後，縦線について上端からも削られていくために生ずる．このような現象を避けるためには，線図形の方向を細線化の過程で検出しなければならない．このため，水平な横線がスケルトンに達する途中の段階で，この例のような横線の存在を交差の左右の部分から検出して，望ましいスケルトンを内挿して決めてしまう処理をしている．また，横線の終点における「うろこ」について，簡単な判断で識別できるものについては，その部分を水平となるように修正している．図3-(b)は，このような修正された細線化処理で得られた例であるが，横線の縦線との持続部におけるスケルトンの歪みがなくなっていることを示している．

細線化処理において，画像の上下および左右の方から一皮ずつ周辺の画素を削り取る各々の処理を1サイクルとすると，スケルトンの画素が決定されるまでのサイクルの数により，元の画像を構成する線画像の太さがわかる．ただし，傾きのある線画像については，実際の太さより大き目のサイクル数を与える．また，端点，交点や角のところでは，細線化処理の手続きに関連した文字の太さと異なった値となっている．元の画像を，このサイクル数をスケルトンの画素に割り当てた画像に変換してその後の処理を進めるが，その画像の一部を**図 4** に示す．この図は，図3の「亜」の真中の口の左上部を拡大したものであるが，サイクル数は16進数で表し，16以上の数は，$g=16$, $h=17$, $i=18$, …としている．

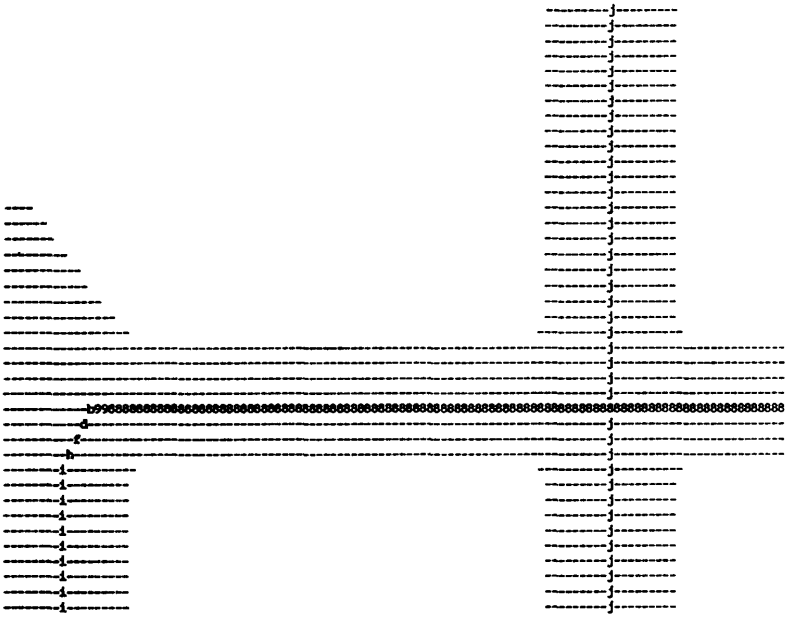


図 4 太さの値で印されたスケルトンのピクセル
Fig. 4 A figure of skeleton pixels marked with the line width.

4. グラフ構造の作成

細線化処理により得られた漢字画像のスケルトンを表現し，解析するための計算機上でのデータ構造として，グラフ構造[9]を用いる．

1つの漢字のスケルトンに対して，端点，交点と角を節(node)とし，節を結ぶスケルトンを枝(edge)に割り当てて，グラフ構造を構成することができる．JIS コード 3021 の「亜」の画像に対して，**図 5** のようなグラフ構造が得られる．さらに角を検出して，矢印で示されているように変形される．本論文において，以後，端点，交点と角を「節点」，それらの間を結ぶスケルトンを「セグメ

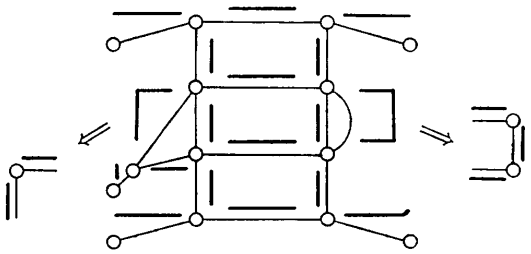


図 5 漢字「亜」に対して作成されたグラフ構造
Fig. 5 A graph structure constructed for the character 亜.

ント」と呼ぶことにする.

節点に対応する計算機上の節には(1)節点の座標値,(2)接続しているセグメントの数およびそれらの形状,(3)接続している節点の節へのポインタを蓄える.ここでセグメントの形状としては,(1)始点と終点の座標値,(2)長さ,(3)水平線に対する傾き,(4)始点から終点に到達するまでの画素ごとの経路を示すチェーン符号化法[10]による方向コード,(5)各々の点での線画像の太さ(前述のサイクル数)とそれに基づく太さの変化(増加,減少など),(6)各々の点での曲率の値などを扱う.

計算機によって,スケルトンのデータから端点と交点の検出をすることは容易である.端点はスケルトンの形状から明らかであり,交点は端点からスケルトンの経路を辿っていき,分岐点に達した時,その場所として検出される.しかし,角の検出にはより複雑な手続きが要求され,多くの計算量が必要である.スケルトン上で隣合った2つの端点または交点によって取り出された1つのセグメントについて,曲率の極大値を持つ点があり,そこで方向の異なる2つのセグメントへ分離できることを見出さなければならない.このためにすべてのセグメントについて曲率を計算する必要がある.ここでの曲率は平均化の概念を導入して,曲率を計算しようとする点の前後の数点の座標値を用いて平均的に求める[9].この画像データによる平均的曲率の実際用いた計算法は,付録として詳しく述べられている.

セグメントの形状を,短い線(4),水平線(4),垂直線(4),右上がりの直線,右上がりの曲線(27),右下がりの直線,右下がりの曲線(27)の7種類に大別する.ここでかっこ内の数字は,以下で説明するようにさらに細分される分類数である.また,短い線とは,曲率を定義するのが難しい長さで,文字幅の 1/36 以下のものとしている.短い線では,水平,垂直,右上がりと右下がりなものの4種に分類する.水平線と垂直線の始点と終点では,図3でも示されているように,「おさえ」や「はね」などの細線化の結果として少々の歪みが現れることがある.この歪みがないものと,始点,終点あるいは両端にあるものの4種に分類する.

曲線においては,曲率は上に凸か下に凸か,曲線の始点あるいは終点での向きが,それぞれ水平または垂直の方向であるか否かなどを組み合わせて9つに分けられ,それぞれに対してさらに,セグメントの両端を結んだ直線の傾きの大きさで,0から30度,30から60度,60から90度の3段階に分類する.

また,水平線では数個所のY座標の平均値を文字要素のY座標値として抽出し,垂直線に対して同様なX座標値を抽出する.ただし,水平方向にX座標,垂直方向にY座標を設定している.

1つのセグメント上で角が検出された際には,そこで2つのセグメントに分解し,各々に対して再帰的に同じような分類を行う.図8で示されている例のように,漢字によっては,もともと別々の2本の横線(CODE 03)が,水平に左右接続していて1本のスケルトンになっていることがある.明朝体の横線の右端に現れる特徴である「うろこ」を検出すれば,1つのセグメントがそこで終っているという判断をする.

セグメントの太さの変化では,長さの方向に沿って一様または単調に増加か減少か,またその変化の大きさについても,適当な3段階に分類して,次の章で説明するセグメントの接続の条件に使用する.太さは交点付近では大きく,端点付近では小さく検出される傾向があるため,セグメントの両端付近での太さは考慮しない.セグメントを5等分し,両端を除く3区間でそれぞれ平均を取り,その3つの値から太さの変化を決定する方法を取っている.

5. セグメント統合

漢字のスケルトンは,これまでセグメントに分解されて扱われてきている.文字要素を抽出するためには,作成されたグラフ構造から,一続きの文字要素から発生した複数のセグメントを検出しなければならない.ここで,表1に示されるすべての文字要素を一度に抽出する方法をとらずに,このうち,曲がりを含んでいる CODE 20, 21, 25～36(これらを以後「複合文字要素」と呼ぶことにする)は,最初の段階では曲

がりの角のところで切り離された形の文字要素で取り出すようにする．これは，文字要素の識別の際に，判定条件が複雑になるのを避けるためである．本論文では，このような文字要素を見出す作業を（セグメント）統合と呼び，文字要素を構成するように，複数のセグメントが統合されたものを「ユニオン」と呼ぶことにする．例えば，1つの横線に他の字画が2個所で交わっている時，左端から最初の交差まで，最初の交差から2番目の交差まで，2番目の交差から終点までの，A，B，C の3つのセグメントに分かれて検出される．その後の処理において，A，B と C を接続させて，それらで1本の横線を形成しているという判断を下さなければならない．

接続している2つのセグメントが統合すべきか否かの判定を行うために，(1)それらのなす角度，(2)形状，(3)太さの変化の3つの特徴を調べる．(1)については，1つのユニオンには曲がりはないとしているので，連続する2つのセグメントのなす角度は180度に近い値であることを仮定している．(2)と(3)によって統合の有無の判断を行うために，「形状の種類による統合ルール」と「太さの変化による統合ルール」を準備する．前者は，2つの水平な線は統合する，右上がりのセグメントと右下がりのセグメントは統合しないなど，前章で述べたセグメントの分類に基づいての判断である．プログラム上では，これらは2つのセグメントの分類を引数とし，統合の可能性を有無を返すサブルーチンとして表現されている．

太さについては，2つのセグメントの接続点に向かい共に太さが減少する場合には，セグメントの統合はしないと考える．2つのセグメントの太さの変化が同一方向連続的である場合でも，接続点付近で太さが急激に変化するものは統合しないと判断する．

図6に示すように，2本の斜めの線が交差する場合や，1つの文字要素に別の2つの文字要素が狭い範囲で接続する場合には，特に短いセグメントが発生する．このように発生した短いセグメントについては，これとその両端に接続しているものの合計3つのセグメントを合わせて，統合ルールを設定する．この場合，1本の右下がりの文字要素に他の文字要素が図 6-(a)のように交差しているか，図 6-(b) のように2つの別々の文字要素が接触しているかの識別は，セグメン

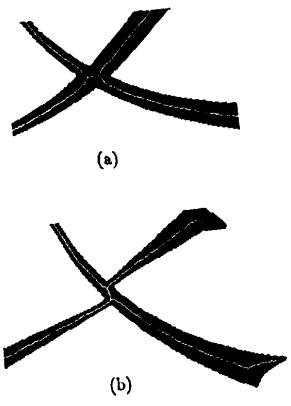


図 6 短いセグメントの発生例
(a) 2つの文字要素の交差，(b) 狭い範囲で1つの文字要素に接触する2つの文字要素

Fig. 6 Examples of generation of the short segments.
(a) The intersection of two character elements, (b) Two character elements closely attached to another character element.

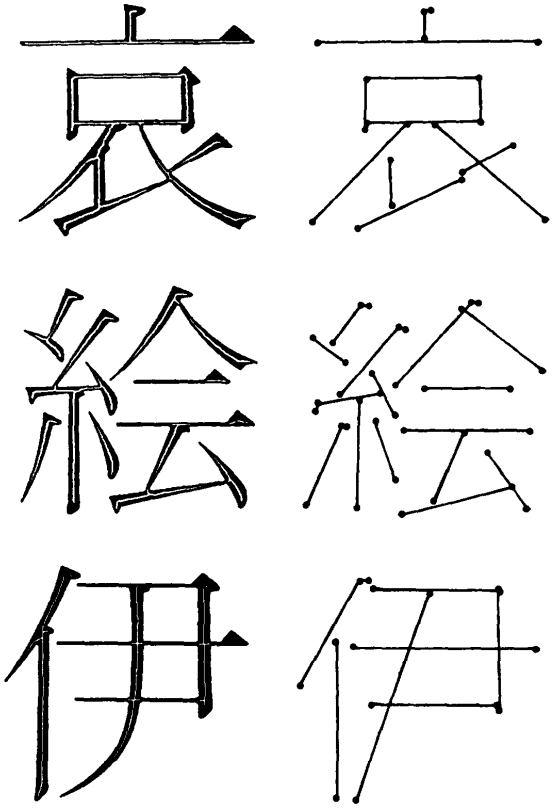


図 7 入力画像(左)に対するセグメント統合の結果(右)得られたユニオンの始点と終点を黒丸で示し，その間を直線で結んだもの．

Fig. 7 Examples of segment arrangements (right side) to form character elements for input images (left side).
The start and end points of unions are shown by bullets and connected by straight lines.

トの太さの変化によって行うことができる．

あるセグメントが複数のセグメントに分岐している場合，まず，2つずつのセグメントの組み合わせをとり，それらのなす角度をこの段階で計算し，統合すべきセグメントの候補を絞る．さらに，セグメントの形状，太さの変化が連続的であることを確認して，統合に最も適する1つのセグメントを選択する．

入力される漢字画像の字画の接触状況や，ノイズのために，誤った統合の判断がなされることがある．また，その形状が詳しく調べられないような短いセグメントが多く含まれる場合には，文字要素の分割されたセグメントを統合していくというこれまで述べた方法ではうまく解決できないことが起きる．このような困難が生じやすい字画の組み合わせとして，「にんべん」や「いとへん」などがある．「にんべん」から得られるスケルトンの例は，図7の中に示されているが，「左はらい」と縦線の接続の所で短いセグメントが生じ，斜めのユニオンとしての統合が成功しないことがある．これらの偏に対応するグラフ構造そのもののマッチングをとることにより，その存在を識別して，セグメント統合を進める方法も組み入れられている．

セグメント統合の結果の例を**図7**に示す．左は入力した漢字画像で，字画の中の線はスケルトンを表す．右はセグメント統合の結果生成されたユニオンの始点と終点を直線で結んだもので，ユニオンの端点は黒丸で示している．

6. 文字要素データの抽出

これまでの実際の処理でユニオンとして検出されるものには，以下のような4種類が含まれる．

1. それだけで1つの文字要素を表すユニオン．
2. これらの複数のユニオンの組み合わせで，最終的な1つの文字要素となるユニオン（複合文字要素の場合）．
3. セグメント統合が不完全なまま生成されたユニオン．1つの文字要素の一部にしか対応しておらず，特殊な場合として，グラフ構造を比較してさらにユニオンの統合を進めなければならない．（例えば，図9における舟へんの真中の「右はらい」など）
4. 「おさえ」や，他の文字要素との接触部に生じた短いユニオン．このユニオンは文字要素の認識に影響がないため無視し，その後の処理の対象にしない．（図7の例でも，「おさえ」に対応した短い線が幾つか見出される．）

これらのことを考慮して，以後，要素組立法に必要なデータを抽出する．

6.1 文字要素の識別

まず，これまで検出したユニオンを1つの文字要素として，対応する文字要素を決める．次にその中から接続している文字要素の組み合わせを調べることにより，複合文字要素に対応するものを選び出す．

CODE 01～19, 22, 24 の 21 種類の文字要素は，検出された1つのユニオンで構成されていて，その決め方は以下のとおりである．まず，すべてのユニオンに対して，ユニオンとしての傾きを調べることにより，**表1**のように4種類に大別する．傾きの計算法では，ユニオンの両端を含むセグメントにおいて，その長さの5分の1だけ内側の点を結んで得られる直線として計算する．これはユニオンの両端において，細線化の際に生じがちな歪みを避けるためである．直線に傾きがあっても，そのユニオンを構成するセグメントの形状として，すべて水平線なら水平ユニオン，すべて垂直線なら垂直ユニオンとする．これは，水平な線や垂直な線がいくつかのセグメントに分かれた場合，ユニオン全体として傾きを持って検出されることがあるからである．この分類に対応する文字要素は，表1の右の欄のようにまとめることができる．

ユニオンの特徴としては，直線性，太さの変化，傾き，他のユニオンとの接続状態，「うろこ」の有無を調べ，さらに，「はね」の検出を行う．「はね」には，CODE 08 のように左へ向かうものと，CODE 20 のように上へ向かうものの2種類がある．「はね」の部分だけで，独立した1つのユニオンとして検出されている場合と，1つのユニオンの一部として含まれている場合とが生じ，その区別のために，太さの変化，長

表1 傾きによるユニオンの分類とそれらに対応する文字要素
Table 1 Classification of unions by slopes and the corresponding character elements.

ユニオンの種類	ユニオンの傾き ($D=dx/dy$)	対応するコード番号
水平ユニオン	$\|D\| \leq 1/15$	03, 04
垂直ユニオン	$D \leq -12$ or $D \geq 15$	06, 08
右上がりユニオン	$1/15 < D < 15$	01, 05, 07, 09～14, 18, 19, 24
右下がりユニオン	$-12 < D < -1/15$	02, 15～17, 22

表 2 右上がりの文字要素の特徴
Table 2 Characteristics of character elements having right upward slopes.

コード番号	太さの変化	その他の特徴
01	∧	曲率が下に凸，傾きの大きさが 60 度以上
05	≡	傾きの大きさが 15 度以下の直線
07	≡	傾きの大きさが 15 度以上の直線
09	≡	左への「はね」，「はね」の始点までは直線
10	∨	曲率が下に凸
11	∨	縦線の途中に接続する始点を持つ
12	∨	直線，曲率が上に凸
13	∨	傾きの大きさが 15 度以下
14	∨	上 1/3 がほぼ垂直
18	∧	直線，長さがフォント幅の 1/3 以上
19	∧	最下位の交点に 3 つのセグメントが接続
24	≡	左への「はね」，始点の左側に横線が接続

表 3 右下がりの文字要素の特徴
Table 3 Characteristics of character elements having right downward slopes.

コード番号	特　　徴
02	曲率が上に凸，他の文字要素が接続しない
15	曲率が下に凸，傾きの大きさが 20 度以上
16	傾きの大きさが 20 度以下，「しんにょう」の一部
17	傾きの大きさが 20 度以下，左払いに接続する始点を持つ
22	上への「はね」を持つ

さ，曲率を利用する．

水平ユニオンでは，右端での他の文字要素との接続，および「うろこ」の有無により，CODE 03 と CODE 04 を区別する．垂直ユニオンでは，左への「はね」の有無により CODE 06 と CODE 08 を区別する．また，右上がりユニオンと右下がりユニオンでは，それぞれ**表 2**，**表 3** に示すような特徴を検出することにより，対応する文字要素を識別している．

CODE 23 は，これまでの処理で右下がりユニオンと右上がりユニオンとに分かれて検出されるが，これに相当する統合ルールを設定せず，構成されるグラフ構造のマッチングを直接調べることにより識別している．

このような方法によって，すべてのユニオンに 1 つの文字要素を当てはめた後，さらに**表 4** に示される組み合わせで複合文字要素の照合を行い，最終的な文字要素を決定する．

表 4 複数の文字要素の組み合わせで構成される（複合）文字要素
∩記号は必要な組み合わせを表す．
Table 4 Character elements constructed by combinations of 2 or 3 character elements. The mark ∩ means the necessary combinations of the elements.

コード番号	文字要素の組み合わせ（数字はコード番号）
20	(06 or 15) ∩04∩（上への「はね」）
21	07∩04∩（上への「はね」）
25	(06 or 15) ∩04
26	03∩06
27	03∩08
28	03∩22
29	03∩15
30	03∩20
31	03∩24
32	03∩(10 or 14)
33	03∩12
34	03∩07
35	03∩21
36	(32 or 33) ∩（左への「はね」を持つユニオン）

6.2 文字要素の代表点と太さ

先に述べた代表点の座標値は，これまでに得られたユニオンに対する情報から容易に抽出できる．文字要素の始点や終点は，スケルトンの端点や交点の位置から取るため，元の輪郭部から少し内側になる．このずれの大きさは，1 つの文字要素であれば同じであるから，最終的には，補正することが可能である．

水平と垂直の文字要素に対しては，そのユニオンを構成するセグメントのXおよびY座標を利用して位置の補正を行う．すなわち，CODE 03 と CODE 04 として検出された場合，始点と終点のY座標は本来同じであるので，実際少々ずれていても，これらの値を揃え，同じように CODE 06 ではX座標を揃える．CODE 11 は「木へん」における左はらいであるが，スケルトンでは始点の位置にずれが生じている（図 8 を参照）．しかし，この文字要素が検出されれば，スケルトンとして望ましい形が前もって分かっているので，その正しい始点を決めることができる．また，縦線の「おさえ」による始点のずれも補正される．

文字要素の太さとしては，水平な線（CODE 03,

CODE 04) と垂直な線 (CODE 06, CODE 08) では数個所のサイクル数の平均値を取り，それ以外の文字要素ではそれらの最大値とする．この際，交点や端点付近の太さはその後の取扱いが複雑になるので特に考慮しないようにし，CODE 03 と CODE 06 を組み合わせた CODE 26 のような文字要素では，構成しているそれぞれの部分的な要素の太さを抽出する．

6.3 他の文字要素との接続の有無

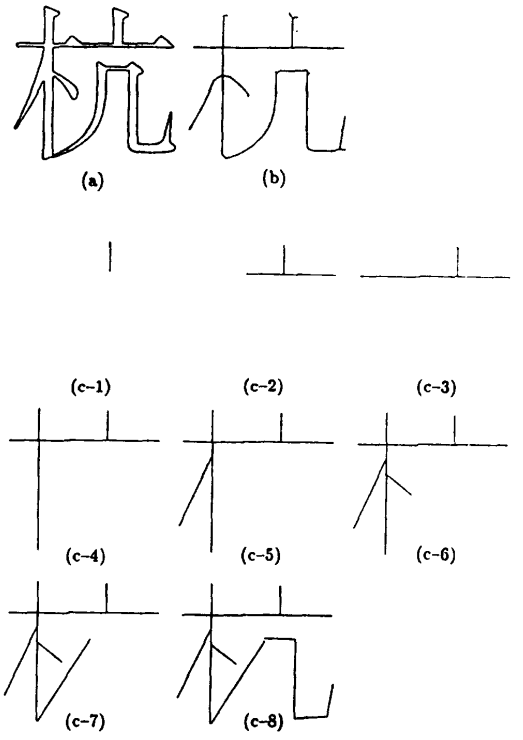
文字要素の両端においての他の文字要素との接続情報が漢字の再構成の時に必要になることは，2章で指摘している．これらについては，両端ともに接続あり，またはなし，始点のみ，または終点のみ接続ありの4つの状態として保存する．図8では，これらを FLAG-0〜3 の符号で区別している．これらのデータは，その文字要素を含むグラフ構造を調べることにより得られるが，前述の短いユニオンが両端に接続している場合は，接続なしとする．また，CODE 06 または CODE 07 について，その下端に CODE 15〜17 が接続している場合(「しんにょう」など)，終点は接続なしとする．

7. 文字要素の認識結果

本方法のプログラムは，すべてC言語で記述し，研究室のワークステーション上で実行した．使用した漢字画像は，明朝体漢字フォント作成用で，画素数の大きさとして 384×384 のものである．文字要素の認識結果の出力例として，JIS コード 393A 「杭」における認識結果を**図8**に示す．この図において，(a)は元の画像の輪郭線，(b)はそのスケルトン，(c)は順々に抽出された文字要素の代表点を直線で結んで得られた線画，(d)は決定された文字要素の番号 (UNION) と種類 (CODE)，始点・終点での他の文字要素との接続状態 (FLAG)，太さ (WIDTH)，代表点の座標値をそれぞれで示す．太さと座標において，0は空の値を示す．この例で，本来の字の形状としては意味を持たないような字画の接触があっても，文字要素の認識が正しく行われていることを示している．

JIS 第一水準 2965 個に本方法を適用したところ，抽出すべきデータ (文字要素の種類，代表点の位置，太さ，他の文字要素との接続の有無) すべてに対して正しく得られた漢字の個数は 2291 であり，認識率は 77.3%であった．

誤った結果の出力例として**図 9** に示す．図 9-(a)では，衣の部分で，鍋蓋の短い縦線とその下の「左はらい」の部分のセグメントを誤って統合している．この場合は，縦線が短いことと「右はらい」がさらに加わって接続点で条件が複雑になったことが誤りの原因




```
UNION : 0    CODE : 06    FLAG : 1    WIDTH : (22) (0) (0)
    (242,33)   (242,102)   (0,0)   (0,0)   (0,0)
UNION : 1    CODE : 03    FLAG : 0    WIDTH : (8) (0) (0)
    (156,102)   (362,102)   (0,0)   (0,0)   (0,0)
UNION : 2    CODE : 03    FLAG : 0    WIDTH : (8) (0) (0)
    (18,104)   (156,104)   (0,0)   (0,0)   (0,0)
UNION : 3    CODE : 06    FLAG : 0    WIDTH : (20) (0) (0)
    (85,29)   (85,357)   (0,0)   (0,0)   (0,0)
UNION : 4    CODE : 11    FLAG : 2    WIDTH : (13) (0) (0)
    (85,135)   (11,297)   (0,0)   (0,0)   (0,0)
UNION : 5    CODE : 02    FLAG : 2    WIDTH : (9) (0) (0)
    (85,173)   (143,218)   (0,0)   (0,0)   (0,0)
UNION : 6    CODE : 14    FLAG : 1    WIDTH : (13) (0) (0)
    (207,163)   (88,357)   (0,0)   (0,0)   (0,0)
UNION : 7    CODE : 30    FLAG : 2    WIDTH : (9) (20) (18)
    (207,157)   (278,157)   (278,343)   (354,338)   (366,261)
```

(d)

図 8 漢字「杭」に対する文字要素データの抽出
(a) 入力画像の輪郭線，(b) スケルトン，(c) 次々と抽出された文字要素の始点と終点を直線で結んだもの，(d) 文字要素の種類，両端での他の文字要素との接続，太さ，等を示す数値記号

Fig. 8 Data of the character elements extracted for the character 杭.
(a) The outlines of input image, (b) The skeletons, (c) Successively extracted character elements drawn by the lines connecting their start and end points, (d) Numerical symbols expressing the classification of the character elements, connection with other character elements at both the end points, and line width.

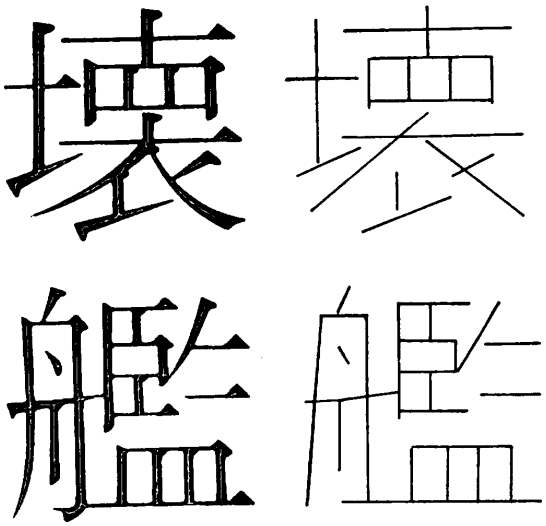


図 9 入力画像(左)に対する誤りのある文字要素の抽出(右)の例
右は抽出された文字要素の始点と終点を直線で結んだもの.

Fig. 9 Examples of erroneous extraction (right side) of character elements for input images (left side).
The start and end points are connected by straight lines.

である．図9-(b)では，皿の部分の下の横線において，左側の「舟へん」のはねと接触し，横線が左に伸びたように抽出されている．この場合は，「はね」の先と横線の右端とを結ぶ直線が水平となったこと，はねの部分の曲り（曲率）がうまく検出されなかったことなどの条件が重なったためである．この例のように，誤認識のあった漢字のほとんどはセグメント統合の段階での失敗によるものであり，認識率の向上のためには，セグメント統合ルールをよりきめ細かくする必要がある．なお，使用したワークステーション(CPU MC 68020, 20 MHz)での平均処理時間は，1文字当たり約120秒で，そのほとんどは細線化の処理時間である．

8. おわりに

現在のシステムの77%という認識率は十分高いとはいえないが，このままでもかなり有用であるといえる．処理の途中結果をディスプレイ端末に表示し，それが正しいかどうかを人間が判断して，もし誤っていれば直ちに修正できるように動作するプログラムも用意している．これはこの研究の本来の目的ではないが，実用になる完成したデータを得るため止むを得ない手段である．本来の自動的な処理を行った後，誤ったものに対して，このような対話的な処理を実行すればよい．ほとんどの場合，セグメント統合の過程での1個所またはせいぜい2個所において指示を与えれば，後は自動的に正しい結果が得られるので，対話的処理の作業の負担はかなり軽減されたものになる．

1つの文字要素について，大きさと位置だけでは文字の違いによる微妙な変化に対応できないということも考えられるが，その場合は文字要素のスケルトン上で，適当な間隔でその座標を保存するようにすればよい．

他の書体に対しては，その書体の特殊な条件を個々のセグメントの分類に反映させなければならないが，主要な処理の方法や流れとしてはこのままで対応できるものと思われる．また，大きさや相対的な位置の少々の違いを除けば，グラフ構造については，書体が変わっても同じである．したがって，明朝体で完全なグラフ構造を製作しておき，他の書体に対する認識処理の際に，これを一種の辞書として利用することは，認識率の向上に役立つものと予想される．

謝辞 本研究に際しまして，いろいろと熱心にご討論頂きました同じ研究室の長谷川武光助教授，佐藤義雄助手に深く感謝いたします．

参考文献

1) Coueignoux, P.: Character Generation by Computer, *Comput. Gr. Image Process.*, Vol. 16, pp. 240-269 (1981).
2) 上原徹三，国西元英，下位憲司，鍵政秀子，菊池純男：ストローク種別に基づく漢字形状生成方式，情報処理学会論文誌，Vol. 31, No. 2, pp. 209-218 (1990).
3) 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. J 72-D-II，No. 9, pp. 1423-1431 (1989).
4) 奥村彰二，佐藤義雄：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，情報処理学会論文誌，Vol. 29, No. 12, pp. 1101-1107 (1988).
5) Pavlidis, T.: *Algorithm of Graphics and Image Processing*, Chapter 9, Computer Science Press (1982).
6) Kwok, P. C. K.: A Thinning Algorithm by Contour Generation, *Comm. ACM*, Vol. 31, No. 11, pp. 1314-1324 (1988).
7) 鳥脇純一郎：画像理解のためのディジタル画像処理(II)，3章，昭晃堂 (1988).
8) 木本伊彦，安田靖彦：分岐歪みのない細線化処理，第12回画像工学コンファレンス報告書，pp.

67-70 (1981).
9) 長尾 真：パターン情報処理，3章および5章，コロナ社 (1983).
10) Freeman, H.: Computer Processing of Line-Drawing Images, *Comput. Surv.*, Vol. 16, No. 1, pp. 57-97 (1974).

付録 曲率を利用した角の検出

一般に，$y=f(x)$ により表される曲線上の点 (x, y) における曲率 R は式(1)で与えられる．

$$R=\frac{d^2y/dx^2}{(1+(dy/dx)^2)^{3/2}} \quad (1)$$

しかし，スケルトンは離散的な点列で与えられているから本質的に折線であり，折線は45度ごとの離散的な方向を取るため，式(1)の微分値を左右の隣接点だけで近似計算しても良い値は得られない．そこで平均化の概念を利用して，近傍の k 個の点列について平均的に考える．曲率を求めようとする点 (x_i, y_i) の左右 k 個の平均を考えた時の微分値を次のように計算する[9]．

$$\frac{d^2y}{dx^2}=d_+-d_-, \quad \frac{dy}{dx}=d_\pm \quad (2)$$

$$d_-=\frac{1}{k}\sum_{j=-k+1}^{0}\frac{y_{i+j-1}-y_{i+j}}{x_{i+j-1}-x_{i+j}} \quad (3)$$

$$d_+=\frac{1}{k}\sum_{j=0}^{k-1}\frac{y_{i+j}-y_{i+j+1}}{x_{i+j}-x_{i+j+1}} \quad (4)$$

$$d_\pm=\frac{1}{k}\sum_{j=-k/2}^{k/2}\frac{y_{i+j}-y_{i+j+1}}{x_{i+j}-x_{i+j+1}} \quad (5)$$

ここで，d_- は左側の勾配の平均値，d_+ は右側の勾配の平均値，$d_\pm$ は左右半々の中央における勾配の平均値である．

これらの式を利用して，セグメントの $(k+1)$ 番目から（セグメント長$-k-1$）番目までの各点について曲率を求めるが，式(3)～(5)の計算において分母の値が0になることがある．この時は垂直線であるからその分数の値を十分大きな値として 1000 とする．次の場合は曲率として特別な値を割り当てて後の処理に用いる．

垂直線	d_- と d_+ が共に900以上	曲率$=-1$
水平線	d_- と d_+ が共に0	曲率$=-2$
斜線	$d_-=d_+$ で上記以外	曲率$=-3$

このようにして求めた曲率の値は k の選び方によりかなり変わるが，角の点ではその周囲の点列の曲率に比べかなり大きな値になり角を検出することができる．また，上記の値 $(-1, -2, -3)$ の数を調べることによりセグメントの直線性がわかり，セグメントの形状の種類決定に役立つ．本研究では k の値を4として曲率を計算し，次の条件を満たす時にその点を角と判断している．ただし，しきい値等の数値は，用いた 384×384 のサイズの漢字フォントにおいて経験的に設定したものである．

1. 基本的には曲率60以上の点を角とする．
2. 近傍(以下近傍とはフォントサイズの12分の1の距離の意味で用いる）に交点がある場合は，角ではない．ただし，近傍に端点がある場合は角とする．

   交点付近ではスケルトンの歪みのために曲率が大きくなる点が現れるため，これを角とはしない．端点付近では文字要素の「おさえ」の部分のスケルトンのために角ができ，この点ではセグメントを分割して，「おさえ」の部分のスケルトンを切り放す．
3. 曲率60以上の点でセグメントを2つに分割した場合，2つのセグメントの形状が共に右下がり，または右上がりのセグメントになる場合は角ではない．
4. 上記の 1., 2. または 3. を満足していても，曲率が100以上なら角とする．ただし，その点の近傍にそれ以上に大きな値の曲率を持つ点があるなら，角ではない．

   角が尖っていない場合は，曲率が大きい点が数点集まるため，曲率最大の点を角とする必要がある．
5. 曲率60以上100未満の点で2つのセグメントに分割した場合，セグメントの形状がそれぞれ右上がりのセグメントと右下がりのセグメントになる場合は角である．

   右側に凸のセグメントや左側に凸のセグメントの形状は設定していないため，2つのセグメントに分割する必要がある．
6. 垂直線における曲率(−1)と水平線における曲率(−2)を含み，−1から−2に移るまでのピクセル数がセグメント長の3分の1以下なら，−1から −2 に移る区間での曲率最大の点（曲率 60以下でも良い）を角とする．

   垂直線から水平線に変化する点は角である．この場合は，角が尖っていないとき曲率が小さくなることがあるため，曲率が60以下でも構わない．

このような条件により使用した漢字画像に現れる角

はほとんど検出できた．しかし，時には角であっても，曲率の値があまり大きく求まらない場合も出てくる．曲率の値が大きくならない角は，細線化の結果として緩やかなカーブになってしまうところであり，その時の対策として，6. の条件（必ず 90 度になる場合）が加えられている．

（平成2年 5 月10日受付）
（平成2年10月 9 日採録）

奥村 彰二（正会員）

1939年生．1962年京都大学工学部原子核工学科卒業．1964年同大学院修士課程修了．1966年同大学院博士課程中退．以降東京大学原子核研究所，高エネルギー物理学研究所勤務を経て，現在福井大学工学部情報工学科に勤務．理学博士．コンピュータグラフィックス，図形・文字入出力システム，パターン認識・理解，人工知能言語に興味をもつ．電子情報通信学会会員．

前田 正弘

1965年生．1988年福井大学工学部情報工学科卒業．1990年同大学大学院修士課程情報工学専攻修了．現在，オージー情報システム総研(株)教育事業部に勤務．CAI システム，パターン認識，知識情報処理等に興味をもつ．