続け字と崩し字に対応したヒューリスティックなストローク合わせ法によるオンライン手書き漢字認識†

大森 健児††

続け字，崩し字を含む手書き漢字をオンラインで認識するための新しい手法を述べる．楷書体の参照パターンからは，続け字，崩し字に対応できるようにするため，隣合うストロークを続けることにより，これらの書体に対するストロークを生成し，それぞれのストロークの特徴を手書き漢字の変化に対応できるようにするため，曖昧性を有するファジィ化データで表現し，それぞれの参照パターンに対してこのファジィ化データから認識のためのプロダクションルールを作成する．入力パターンも同様にそのストロークの特徴をファジィ化データで表し，プロダクションルールを起動してファジィ推論を行い最も高い確信度を与える参照パターンを認識結果とする．このとき適用するプロダクションルールを少なく抑えるために，入力ストロークと参照ストロークの対応関係をヒューリスティックに求め，各参照パターンに対して仮説を設定し認識を行う．実験結果によれば，正しく認識した割合は楷書体で 95.2%，行書体で 75.6% であった．また，5位までで正しく認識した割合はそれぞれ 99.5%，93.6% であった．また，認識時間は 32 ビットマイクロプロセッサ 80386 (20 MHz) で，「雑」の楷書体に対して 2.2 秒，行書体に対して 35.3 秒であった．

1. はじめに

高度情報化社会が進む中で，人間と機械の間のインタフェースはますます重要になってきている．特に，多種類の文字を使用している文化圏にあっては，文字の入力は，以前にも増して重要な課題である．日本語の入力においても同様で，これまでにさまざまな対策が講じられてきた．その結果，ローマ字漢字変換，平仮名漢字変換等の開発によって相当に改善された．しかし，情報機器に馴染んでいない人々にとって，あるいは，キーボードを取り付けることができないような機器では，日本語の入力は依然として大きな問題である．さらに，情報化社会の進展にともなってこの問題は拡大しつつある．

日本語の入力に対する有効な解決手段の1つは，手書き文字をコンピュータに入力し，それを認識させるものである．これには，文字を書いているときの筆の動きに関する情報を得て行うものと，紙などの上に既に書かれてしまっている文字についてそのイメージ情報を得て行うものとの2つの方法がある．後者に対しては，数字を始めとする比較的文字種の少ないものに対して，郵便番号の自動認識などで実用化が進んでいる．しかし，文字種の多いものについては未解決である．一方，前者，すなわち，オンラインでの手書き文字認識においては，正しい筆順で書く，あるいは，正しい画数で書くなどの制限のもとでは，高い認識率をあげており，実用化もなされている．しかし，このような制限のもとでは，書くときに必要以上に注意を払う必要があるため，その利用範囲はやはり限定されている．このような事情から通常の書き方で書いたものが正しく認識されるようなオンライン認識技術の確立が強く望まれている．

これまでオンラインの手書き漢字認識に対しては，楷書体を対象としたもの[1)~4)]，続け字，崩し字を対象としたもの[5)~9)]などの論文が多数発表されている．手書き文字の認識技術を実用的なものにするためには，続け字あるいは崩し字についても相当の認識率を保つことが要求されるが，これまでの論文では文格的な崩し字には取り組んでいない．崩し字の一つの目安は行書体と思われるが，行書体ではストロークが続けられる箇所の総数の平均は 2.5 前後である．これに対して，最も崩し字に本格的に取り組んでいると思われる論文[7)]でも，ストロークを続けていないもの 61.1%，1箇所だけ続けてあるもの 20.9% にすぎない．このような現状から，行書体程度までの崩し字にも対応できる技術の確立が望まれている．

このような要望に答えるため，本論文では続け字および崩し字を含む手書き漢字に対する新しいオンライン認識方法の提案とその実験結果について報告する．

手書き漢字の認識方法に対しては，DP マッチング法[1)]や構造解析法[2)]などさまざまな方法が開発されてきた．周波数領域での特徴点の抽出法[10)~12)]もまたそ

† On-line Handwritten Kanji Recognition by Heuristic Stroke Matching Approach by KENJI OHMORI (Department of Industrial Engineering and Systems Engineering, Faculty of Engineering, Hosei University).

†† 法政大学工学部経営工学科

の1つである．論文[4]は周波数領域での特徴点に曖昧性を持たせることで，楷書体での手書き漢字認識がファジィ推論により効果的に行えることを示している．この手法は手書き漢字の曖昧さを利用したもので認識率も高く有力な方法と考えられる．しかし，この方法は正しい画数で入力された手書き漢字という制限のもとに提案された方法である．続け字や崩し字にも対応できるようにするためには，続け字あるいは崩し字の自動生成，処理量の少ない認識方法の確立といったような問題を解決する必要がある．

また，続け字や崩し字を対象とする場合には，ストロークの結合箇所が組み合わせ的になるため，処理量が楷書体のそれと比較すると膨大になる．このため，効率的にストローク合わせを行うことが必要となるが，論文[7]での選択的ストローク結合はある程度の成果を挙げているものの，認識率からみると十分なものとはいえない．このため，続け字や崩し字を対象とする場合には，認識率をさほど低減させることなくストローク合わせを効率的に行うことが必要である．

この論文では，①隣合うストロークの結合による続け字や崩し字の自動生成，②参照パターンからこれらの文字に対応したプロダクションルールの作成，③プロダクションルールを用いての入力パターンのファジィ推論による認識，④適用するプロダクションルールを抑えるための仮説の設定，⑤入力パターンと参照パターンのストロークをヒューリスティックに合わせる方法，などの技術を確立し従来の問題を解決した．そこで，これらについて以下で説明する．

2. ファジィ推論による認識アルゴリズム

手書き文字は，同じ人であってもその都度文字は少しずつ変形する．また，人が変われば，相当に異なることもある．このため，完全に一致する参照パターンを認識結果として選び出すという方法は手書き文字の認識ではとることができない．このため，入力パターンに最も似ていると思われる参照パターンを選び出すということが手書き文字認識では必要となる．そこで，ここでは文字の特徴を曖昧な表現で表し，これを用いて推論することを試みた．すなわち，参照パターンの特徴をストロークの周波数領域でのファジィ化データで表し，これより，プロダクションルールを生成する．また，入力パターンも同様にファジィ化データに変換し，プロダクションルールを適用して認識するという方法をとった．

ファジィ推論によるオンライン手書き認識のアルゴリズムは，参照パターンよりプロダクションルールを作り出すまでの前準備と，このプロダクションルールを用いて入力パターンの認識をオンラインで行う部分とに分かれている．参照パターンよりプロダクションルールを作り出す部分は次のようになっている．①参照パターンを正しい筆順で楷書体で入力する．②隣合うストロークを接続することにより続け字や崩し字に対応したストロークを生成する．③これらのストロークをファジィ化データに変換する．④プロダクションルールを生成する．これらの処理は認識のシステムを作るときに一度だけ行うもので，その後ではこの作業を行う必要はない．

実際のオンラインの漢字認識は前準備の段階で作成されたプロダクションルールを利用して次のように行う．①タブレットより入力されたパターンをファジィ化データに変換する．②それぞれの参照パターンに対するプロダクションルールを取り出す．③ファジィ化された入力パターンにプロダクションルールを適用し，この参照パターンに対する確信度を得る．④最も高い確信度を与える参照パターンを認識結果として出力する．

以下ではこれらについて詳しく説明する．

イ）ファジィ化データ

参照パターンはファジィ化データの形でシステムに登録するが，これは次のように行う．

① データの入力　参照パターンの登録は，参照パターンとして用いる漢字をタブレット上に，正しい画数と正しい書き順でかつ楷書体で書くことにより始まる．タブレットからはストロークを構成する座標点列がコンピュータに入力される．

② データの正規化　文字の大きさを一定にするために，タブレットより入力された参照パターンを，256×256 ドットの正方形の枠の中にちょうど納まるように，縦と横それぞれ別の縮尺で拡大あるいは縮小する．このとき，座標系は左上を原点とし，x 軸は右のほうにまた y 軸は下のほうに行くに従って増加するものとする．

③ フーリエ変換　ストロークに対して座標系ごとに高速フーリエ変換を用いてフーリエ変換を行う．このため，ストロークを始点と終点を含む 2^n の等間隔の座標点の列で表す．（現在のシステムでは $n=3$ である．）この後，この座標点列から x 座標系と y 座標系での点列を得，座標系ごとに点列に対して高速フーリ

エ変換を施し，フーリエ級数展開したときの各係数の値を得る．

④ 量子化　ファジィ化データの表し方はいろいろ考えられるが，ここでは，係数の値を0から15までに量子化し，それを利用する．量子化は，量子化の大きさ，量子化の中心点，移動量と呼ぶ3つの変数 β, γ, δ を用いて，係数の値 α に対して次のように行う．

$a=\mathrm{DIV}\ ((\alpha+\gamma), \beta)+\delta$

ただし，a が15を越えるときはこれを15とし，0を下回るときはこれを0とする．

ここで，DIX (x, y) は x を y で割ったときの商（整数）である．係数の値が0のとき量子化値が，第0係数では0になるように，第1，第2係数では8になるよう，変数の値は次のように定めた．

$\beta=32$, $\gamma=0$, $\delta=0$　第0係数に対して

$\beta=16$, $\gamma=8$, $\delta=8$　他に対して

⑤ ファジィ化データへの変換　ファジィ化データは量子化した値 a を用いて［a］で表す．これは，およそ a であるという意味を持ち，一般にファジィ数と呼ばれるものである．したがって，［a］は a から少し離れた量子化値である可能性も有している．それぞれの値に対してどの程度の可能性を有しているかをメンバシップ値で表すが，ここでは，**図1**に示すように，ファジィ化データ［a］に対するメンバシップ値は，量子化値 a でメンバシップ値1を持ち，量子化値が1下がるにしたがって，メンバシップ値は0.2下がるものとした．

ストロークの特徴は，これらに加えて，ファジィ化されたストロークの長さおよび y 軸方向での移動量と x 軸方向での移動量との比で表した．

ロ）続け字および崩し字への対応

参照パターンは先に述べたように楷書体で与えられる．このため，続け字および崩し字に対応できるようにするためには，楷書体の文字からこれらの文字を作り出す必要がある．これらの字体は，楷書体では離して書くいくつかのストロークをなめらかに続けて書くことによりできあがっている．続け字あるいは崩し字は，楷書体での筆の運びを踏襲しながら作られるものであるため，続けて書くストロークは楷書体での筆順においては隣合っているものがほとんどである．

そこで，この性質を利用して参照パターンから続け字や崩し字に対応するパターンを次に述べる方法により作り出すこととした．楷書体での参照パターンから書き順の上で隣合っているストロークを取り出し，隣合うストロークの間にそれらを接続するストロークがあると見なして，これらを続け，続け字あるいは崩し字に対応したパターンを作り出す．**図2**に参照パターン「久」とこれを続けることによって得たパターンを示す．続けたストロークが，楷書体でのストロークとその間にストロークを入れたものに完全に一致していないのは，ストロークを等分し，$2^n(n=3)$ の点で表したことによる．

タブレットより入力された参照パターンからは，これまでに述べた方法により，楷書体およびこれらから作り出された続け字あるいは崩し字のストロークに対してその特徴を表すファジィ化データを得る．このファジィ化データは次に述べるプロダクションルールを作り出すためにシステムに登録される．

ハ）プロダクションルール

プロダクションルールは，参照パターンから得たファジィ化データより作り出される．それは楷書体，生成された続け字あるいは崩し字ごとに作り出される．参照パターン「久」に対するプロダクションルールを**図3**に示す．

続け字や崩し字のように曲線の多いストロークを有する文字を対象とする場合には，プロダクションルールは日常使うような表現ではない．しかし，楷書体の漢字のみを対象とする場合には，楷書体でのストロークは，まっすぐかあるいは中央付近で曲がっているものがほとんどである．この場合には，フーリエ級数の

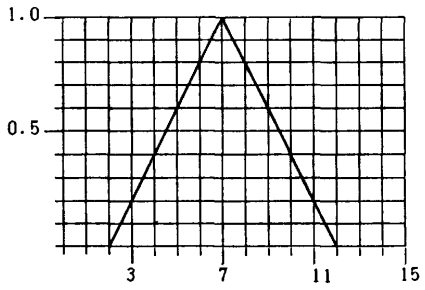


図1 ファジィ化データ［7］に対するメンバシップ関数
Fig. 1 A membership function for fuzzified data [7].

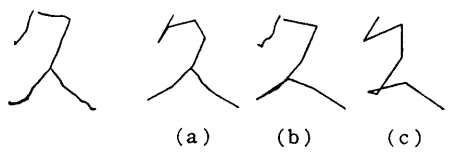


図2 参照パターン「久」とそれから作られた続け字
Fig. 2 A cursive character generated from reference pattern "久".

第0係数がストロークの重心を表すほかに，第1係数は始点と終点の離れ具合いを，また，第2係数は曲がり具合いを示すようになる．したがって，プロダクションルールはこれらの程度を表す言葉を用いて表すことができるようになる．例えば，量子化値の代わりに論文[4]の日常語によるファジィ化データを用いたとすると，ルール久1は図4のようになる．このようにすると，プロダクションルールは感覚的にとらえやすいものとなる．

ニ）確 信 度

プロダクションルールを得るまでの手順は，手書き漢字認識システムを作成するときに必要とするもので，一度だけこの処理を行えば，筆者を特定しない認識システムが実現する．このようなシステムを作成した後では，誰かがタブレット上に漢字を書きさえすれば，これが認識システムに入力され，プロダクションルールを用いての認識が始まる．認識の手順は次のようになっている．

入力パターンを，参照パターンのときと同様の方法で，まず，ファジィ化データに変換する．この後，入力ストロークと同一の画数のプロダクションルールを1つずつ読み出し，このルールの結論に対する確信度を計算する．そして，最も高い確信度を与えるものが認識結果となる．（すべてのプロダクションルールを読み出すことは処理時間の増大を招くので，実際の処理ではこれを減少させる方法を用いる．これについては後で述べる．）

プロダクションルールの結論に対する確信度の計算は次のように行う．

まず，各ストロークに対してその類似度を計算する．これはストロークの特徴を述べているそれぞれの項目に対する一致度を計算し，その最小値をストロークの類似度とする．ストロークは長さ，移動量の比，各座標系での周波数の係数でその特徴を表している

ルール久1
第1ストロークの長さが［1］で，移動量の比が［3］で，x座標系でのフーリエ級数の第0係数が［2］で，第1係数が［10］で，第2係数が［8］で，y座標系でのフーリエ級数の第0係数が［2］で，第1係数が［6］で，第2係数が［8］で，
第2ストロークの長さが［6］で，移動量の比が［2］で，x座標系でのフーリエ級数の第0係数が［6］で，第1係数が［11］で，第2係数が［5］で，y座標系でのフーリエ級数の第0係数が［7］で，第1係数が［0］で，第2係数が［8］で，
第3ストロークの長さが［2］で，移動量の比が［2］で，x座標系でのフーリエ級数の第0係数が［11］で，第1係数が［4］で，第2係数が［8］で，y座標系でのフーリエ級数の第0係数が［13］で，第1係数が［5］で，第2係数が［8］で，
あるとき，入力パターンは「久」である．

ルール久2（参照ストロークの1と2を接続）
第1ストロークの長さが［9］で，移動量の比が［2］で，x座標系でのフーリエ級数の第0係数が［5］で，第1係数が［8］で，第2係数が［5］で，y座標系でのフーリエ級数の第0係数が［6］で，第1係数が［2］で，第2係数が［11］で，
第2ストロークの長さが［2］で，移動量の比が［2］で，x座標系でのフーリエ級数の第0係数が［11］で，第1係数が［4］で，第2係数が［8］で，y座標系でのフーリエ級数の第0係数が［13］で，第1係数が［5］で，第2係数が［8］で，
あるとき，入力パターンは「久」である．

ルール久3（参照ストロークの2と3を接続）
……

ルール久4（参照ストロークの1と2と3を接続）
……

図3 参照パターン「久」に対するプロダクションルール
Fig. 3 The production rules for reference pattern "久".

ルール久1
第1ストロークの長さは非常に短く，移動量の比が等しく，x座標系で見たとき，ストロークの重心が左端に相当に接近していて，終点が始点に対して左に接近していて，曲がり具合いがまっすぐで，y座標系で見たとき，ストロークの重心が上端に相当に接近していて，終点が始点に対して下に接近していて，曲がり具合いがまっすぐで，
……
あるとき，入力パターンは「久」である．

図4 参照パターン「久」に対するプロダクションルール
Fig. 4 The production rules for reference pattern "久".

図 5 入力パターン「久」
Fig. 5 Input pattern "久".

表 1 入力パターン「久」のファジィ化データ
Table 1 The fuzzified data for input pattern "久".

画数	長さ	移動比	x座標系の係数			y座標系の係数		
			0	1	2	0	1	2
1	9	2	6	10	5	7	2	10
2	2	2	12	5	8	13	6	.8

が，これらの項目に対する一致度は次のように計算する．その項目に対するファジィ化データと入力パターンでの対応するファジィ化データとの間で，メンバシップ値を量子化値に沿って連続的に比較し，それぞれに対して小さいほうを選び，最後にこれらの中で最大のものを選んで一致度とする．したがって，ファジィ化データ［a］と［b］との間の一致度は $|a-b|/10$ となる．

各ストロークの類似度を求めた後で，このルールの結論に対する確信度を計算するが，これは類似度の平均値とする．

図 5 に示すような行書体での漢字「久」を入力したとする．このときのファジィ化データは**表 1** のようになる．いま，ルール久2を起動したとする．第1ストロークの x 座標系での第1係数の一致度を計算すると，0.8 となる．他の項目でこの値を下回るものがないので，第1ストロークの類似度は 0.8 となる．同様に第2ストロークの類似度は 0.9 となる．このため，入力パターンが「久」である確信度は 0.85 となる．他のルールでこの確信度を上回るものが現れないとすると入力パターンは「久」と認識される．実際の実験での認識結果は久 (0.85)，火 (0.80)，六 (0.80)，大 (0.75)，欠 (0.75) となった．なお，括弧内の数値は確信度である．

3. 全数探索と仮説設定

続け字や崩れ字まで認識できるようにしようとすると，入力パターンと同一の画数を有するプロダクションルールの数は膨大になる．いま，入力パターンの画数を m とする．また，楷書体での画数が n $(>m)$ である参照パターンとの間で認識を行ったとする．この参照パターンから画数が m であるプロダクションルールは，結合可能な箇所が $n-1$ 個，結合すべき箇所が $n-m$ 個であることを考慮すると，${}_{n-1}C_{n-m}$ 個存在する．例えば，$m=10$，$n=14$ とすると1つの参照パターンに対して 715 個ものプロダクションルールが存在する．

さらに，入力パターンに対しては書き順を制限していないため，入力パターンについては，正しい書き順に並べ換える必要がある．プロダクションルールを適用したときに最も高い確信度を与える入力ストロークの並びが正しい書き順のものであるとすると，すべてのストロークの並び換えについてプロダクションルールを適用することが必要になる．論文[4]の動的計画法により探索する空間を縮小したとしてもその処理量は依然として相当である．

ここではこの問題を解決するために，1つの参照パターンに対しては1つのプロダクションルールのみを適用し，このプロダクションルールに対しては1通りのストロークの並びしか適用しない方法を考える．これは入力パターンと参照パターンとのストロークの対応関係について仮説を設定して行うものであるが，例で説明する．

図5の入力パターンが与えられたものとする．この入力ストロークは「ク」の部分を続けたものである．この入力パターンは個性の強い人によって書かれたため，その入力の順序は通常の方法とは逆に「㇏」が先で「ク」が後であったとする．このとき参照パターンごとに仮説をたてるが，例えば，参照パターン「久」に対しては次のような仮説を設定する．

仮説「久」:

入力パターンが「久」であるとするならば，1番目の入力ストロークは3番目の参照ストロークに対応し，2番目の入力ストロークは1番目と2番目の参照ストロークを続けたものに対応する．

また，参照パターン「欠」に対しては次のような仮説を設定する．

仮説「欠」:

入力パターンが「欠」であるとするならば，1番目の入力ストロークは4番目の参照ストロークに対応し，2番目の入力ストロークは1番目と2番目と3番目の参照ストロークを続けたものに対応する．

このような仮説をもとに推論を行うことにすると，

適用すべきプロダクションルールは各参照パターンに対して1つに抑えられ，また，ストロークの並び順も一意であるので，少ない処理量で各参照パターンに対する確信度を得ることができるようになる．

各参照パターンに対して正しい確信度が得られるかどうかは正しい仮説を立てているかどうかが問題になる．特に，入力パターンと同一漢字に対する参照パターンにおいては正しい仮説を立てていることが要求される．入力ストロークと参照ストロークを正しく対応させるために全数探索を行ったのでは，仮説を設定した意味がなくなる．そこでヒューリスティックにストローク合わせを行うということを考える．

4. ヒューリスティックなストローク合わせ

参照パターンと入力パターンとが同一の漢字に対するものであるとき，対応するストロークは当然のことであるがよく似ている．例えば，「久」に対する参照パターンと入力パターンのそれぞれのストロークを**図 6** に示すが，これからも明らかである．図6で上に位置するストロークは参照パターンに対するもので，下に位置するストロークは入力パターンに対するものである．参照パターンに対しては楷書体でのストロークをその筆順に沿って記した後で，隣合うもの同士を続けることによって得たストロークを記した．また，入力パターンに対しては入力順に記した．以降では楷書体でのストロークを実参照ストロークと呼ぶこととし，隣合うもの同士を続けることにより得たものを続け参照ストロークと呼ぶこととする．これらを特に区別する必要がないときは単に参照ストロークと呼ぶことにする．

ここでは，対応するストロークの間では類似度が高いという例で示した性質を利用してヒューリスティックにストロークを合わせることを考える．まず，各入力ストロークに対して最も類似度の高い参照ストロークを割り当てる．図6の場合には入力ストロークAに対しては参照ストロークcが，入力ストロークBに対しては参照ストロークdが対応することになる．多くの場合にはこの割当で参照ストロークと入力ストロークはうまく対応する．しかし，入力ストロークBにおいて，最初の斜め棒をほとんど書かなかったとするとこの入力ストロークは参照ストロークdではなく参照ストロークbと対応してしまう．この場合には，楷書体を構成する実参照ストロークのうちの1つがどの入力ストロークにも対応しないことになり，ストローク合わせに失敗する．

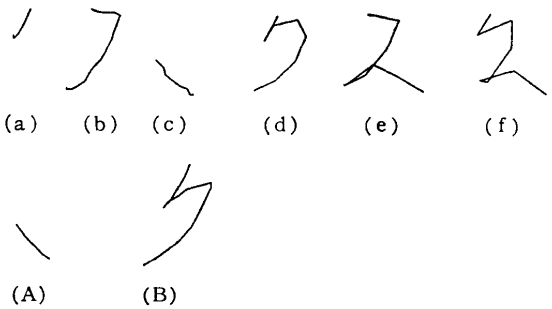


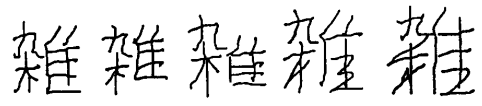
図 6 ヒューリスティックなストローク合わせ
Fig. 6 A heuristic stroke matching.

このように入力パターンのほうに形状のずれあるいは位置のずれが存在する場合には，各入力ストロークに対して最大の類似度を与える参照ストロークを割り当てたとすると，入力ストロークが割り当てられていない実参照ストロークや入力ストロークがだぶって割り当てられている実参照ストロークが存在する．プロダクションルールを適用するためにはすべての実参照ストロークがある入力ストロークの構成要素の全体あるいはその一部になっている必要がある．また，そのような入力ストロークは1つに限られる．そこで入力ストロークが未割当のあるいはだぶっている実参照ストロークについてはこれらを解消する必要がある．ヒューリスティックにストロークを合わせる方法では，各入力ストロークに対して最大の類似度を与える参照ストロークを割り当て，上記の問題が発生している実参照ストロークに対してこれらを解消し，すべての実参照ストロークが1つの入力ストロークの構成要素となるようにする．以下ではこの解消について述べる．

イ）だぶりの解消

解消は複数の入力ストロークが割り当てられている実参照ストロークに対して行う．実参照ストロークに対して複数の入力ストロークが割り当てられる原因は，参照パターンの中に似たようなストロークが複数存在し，入力パターンの中でそれらストロークの中のあるものの位置が他のストロークの位置のほうにずれた場合に起こる．

その例を示す．**図 7**-(a)は，「雑」の参照パターンで，図 7-(b)から(e)はさまざまな入力パターンである．「雑」においては，「ふるとり」と呼ばれる右側の部分に，似たようなストロークとして4つの横棒のストロークが存在する．図 7-(b)に示すようにこれらのストロークを全体的に上のほうにずらして書いた



(a) (b) (c) (d) (e)

図 7 「雑」に対する参照パターン(a)と入力パターン(b)-(e)
Fig. 7 Reference and input patterns for character "雑".

とする．このとき，入力ストロークに対して最大の類似度を有する実参照ストロークを割り当てたとすると，割当関係は**表 2** の b 欄に示すようになる．「ふるとり」は正しい書き順では，左側の斜め棒（7番目のストローク），左側の縦棒（8），右側の斜め棒（9），いちばん上の横棒（10），右側の縦棒（11），以下残りの横棒の順である．入力パターンに対しては，処理の中では書き順に沿った番号をそのストロークに与えるが，ここでは，分かりやすくするために対応すべき実参照ストロークの番号を入力ストロークに与えた．また，続きのストロークに対しては対応すべき実参照ストロークの集まりの中で最も若い番号のものをそれに割り当てた．したがって，これらの入力パターンは必ずしもこれに与えられた番号に沿って入力されたわけではない．

図 7-(b)は楷書体での類似のストロークを上にずらしたものであるが，これからいくつかの関係を知ることができる．ここでは，まずこれについて述べることにする．これは，後で述べるように汎用的なものとなる．

（a） 類似の参照ストロークと入力ストロークの関係

参照パターンの中に，ほぼ縦に一列に並んでいる似たような横棒のストローク $s_{p(1)}, s_{p(2)}, \cdots, s_{p(n)}$ が存在したとする．ここで，$p(i)$ は正しい筆順での順番を表すもので，$s_{p(1)}, \cdots, s_{p(n)}$ はその順番に従って並んでいるものとする．ここで，漢字での筆の運びを考慮して，これらのストロークは上から下に書かれたものであるという仮定をする．このように仮定すると，先の横棒のストロークの並び $s_{p(1)}, \cdots, s_{p(n)}$ は上から順に並べたものでもある．

入力パターンを書くときに，$s_{p(1)}, s_{p(2)}, \cdots, s_{p(n)}$ に対応する入力ストロークを上のほうにずらして書いたとする．このとき，入力ストロークを上から下に向かって順次拾ったとし，それらを $i_{q(1)}, i_{q(2)}, \cdots, i_{q(n)}$ とすると，仮定により $s_{p(i)}$ と $i_{q(i)}$ とが対応すべきものである．なお，$q(i)$ は入力パターンを書いたときの順番を表すものとし，i はこれらの入力ストロークを上から数えたときの順番とする．

表 2 参照ストロークと入力ストロークの初期割当
Table 2 The initial stroke assignment for reference and input strokes.

参照ストローク番号	入力パターン a	b	c	d	e
	1	1	1	1	1
	2	2	2	2	2
	3	3	3	3	2
	4	4	4	4	4
	5	5	5	5	4
	6	6	6		4
	7	7	7	7	7
	8	8	8	7	7
	9	9	9	9	
	10	10 12		10	10
	11	11	11	11	9
	12	13	10 12	12	12
	13	14	13 14	12	13
	14				13

今，それぞれの入力ストロークに対して最大の類似度を与える実参照ストロークを割り当てたとする．似たようなストロークの間では，最も近いところに位置する実参照ストロークが最大の類似度を与える．したがって，入力ストロークを上のほうから順に見てきたとき，これに割り当てられる実参照ストロークは，あるものはだぶって，あるものは飛び越して割り当てられるものの，上のほうから順次割り当てられる．

逆に，この割当を実参照ストロークのほうから見ると次のようになる．

実参照ストローク $s_{p(i)}$ に入力ストローク $i_{q(j)}$ が割り当てられているとする．このとき上のほうにずれていることから，$l \leq j$ である．また，$i_{q(j)}$ より上のほうにある入力ストローク $i_{q(i)}$ は，$s_{p(i)}$ かあるいはそれより上に位置する実参照ストロークに割り当てられている．先の仮定から，上のほうにある実参照ストロークは正しい筆順では先に書く．このため，$i_{q(i)}$ に割り当てられた実参照ストロークを $s_{p(k)}$ とすると，$i<j$ のとき，$k \leq l$ である．

（b） だぶり解消の手順

入力ストロークに対して最大の類似度を与える実参照ストロークを割り当てたものを初期状態とし，だぶりの解消は以下のように行う．

① 複数の入力ストロークが割り当てられている実参照ストロークを探し出す．この実参照ストロークを

$s_{p(i)}$ とする．そして，これに割り当てられている入力ストローク $i_{q(k)}, i_{q(k+1)}, \cdots, i_{q(k+m)}$ を取り出す．（これらの入力ストロークは上から順番に並べたとき連続して並び合っているものである．）次に，これらの中で，最も下側に位置する入力ストローク $i_{q(k+m)}$ を探し出し，これを $s_{p(i+1)}$ に割り当てる．

最も下側に位置する入力ストロークは次のようにして探し出すことができる．実参照ストローク $s_{p(i)}$ より筆順が後で，入力ストロークが未割当の実参照ストローク $s_{p(j)}$ を取り出す．だぶっている入力ストロークの中で，最も下側に位置する入力ストロークは，この未割当の実参照ストロークに最も近いところに位置する．このため，この実参照ストロークとこれらの入力ストロークとの類似度を比較すると，最も下側に位置する入力ストロークが最大の類似度を与える．したがって，これらの入力ストロークの中で，最大の類似度を与える入力ストロークを探せばよいことになる．（ただし，形状のずれが大きい場合には誤った割当を行う場合がある．これはヒューリスティックに処理することの限界である．）

② ①の操作を複数の入力ストロークが割り当てられている実参照ストロークがなくなるまで繰り返す．

（c） 手順の正当性

入力ストロークと実参照ストロークが正しく対応するようになることは，以下の議論で保証される．入力ストローク $i_{q(i)}$ に対して，現在割り当てられている実参照ストロークを $s_{p(k)}$ とする．$i_{q(i)}$ が正しく対応すべき実参照ストロークは $s_{p(i)}$ であるので，入力ストロークが本来あるべき位置からどれだけ離れているかを，$|i-k|$ で表すことにする．各入力ストロークに対してこれらを求め，その総和を非照合度とする．非照合度が0であるとき，入力ストロークと実参照ストロークは正しく対応しているといえる．先の操作で非照合度は単調に減少するので，正しい対応関係に導くといえる．（ただし，これも形状のずれがそれほど大きくなく，類似性を支配しているのは位置ずれという場合である．）

だぶり解消を先の図 7-(b)について行うと次のようになる．この例では似たような実参照ストロークは s_{10}, s_{12}, s_{13}, s_{14} である．一方，これに対応する入力ストロークを上から下に順に拾ってくると，i_{10}, i_{12}, i_{13}, i_{14} となる．そして，添え字の同じものが対応すべきストロークとなる．類似度で割当を行うと先に示した表2のb欄のようになる．ここで，s_{10} に対して i_{10} と i_{12} がだぶっているので，入力ストローク未割当の s_{14} との類似度を計算する．この結果，i_{12} のほうが高い類似度を与えるので，これを s_{12} に割り当てる．今度は，s_{12} がだぶるのでこの操作を再び繰り返す．このような操作を続けることにより，最後に正しい割当を得る．

（d） 汎 用 性

これまではほぼ縦に一列に並んでいる横棒のストロークを上のほうにずらして書いた場合について説明したが，下のほうにずらした場合は操作の方向を逆にすることによりだぶりを解消することができる．また，ほぼ横一列に並んでいる縦棒の似たようなストロークについては右あるいは左にずらして書いたりするが，これも同様に解決することができる．さらに斜め棒についても同様である．

「雑」の「ふるとり」の部分には似たようなストロークとして，いままで説明してきた横棒のストロークのほかに，縦棒と斜め棒のストロークがある．このため，横棒を上のほうに，縦棒と斜め棒を左のほうにずらして書くなどということも起こるが，このだぶりもそれぞれの似たようなストロークについて先に述べた方法を適用することにより解決することができる．

入力ストロークをだぶらせる要因には，似たようなストロークの集まりが一方にずれる場合のほかに，ある方向に縮小する場合と，ある方向から拡大する場合とが考えられる．図 7-(c)は「ふるとり」の部分を上下に縮めて書いた場合で，その割当関係は表2のc欄のようになる．縮小あるいは拡大によって入力ストロークにだぶりが生じた場合には，その中心で似たようなストロークの集まりを2分割すると，それぞれは一方にずれたものとして扱うことができる．このため，先に述べた方法によりだぶりを解消することができる．

これまでは議論を簡単にするために，楷書体で書いた入力パターンでの入力ストロークのだぶりについて説明してきたが，続け字あるいは崩し字においてもこの議論は同じである．楷書体においては，似たような実参照ストロークが存在しているときということで議論を進めてきたが，続け字あるいは崩し字を含めてこの議論を一般的に行うためには，実参照ストロークを参照ストロークに置き換えて議論を行えばよい．ただし，似たような続け参照ストロークが表れる可能性は少ない．

ロ）未割当の解消

だぶりの入力ストロークを解消した後でも，入力ストローク未割当の実参照ストロークが存在する場合がある．これは次のような場合に起こる．入力パターンが続け字あるいは崩し字である場合には，続けてあるいは崩して書いた入力ストロークには，複数の実参照ストロークが対応する．このとき，一部の実参照ストロークの集まりがこの入力ストロークに対して最大の類似度を与えてしまうと，残りの実参照ストロークが未割当のものとして残される．

この残り方は，実参照ストロークの集まりが筆順の上で続いているか否かによって，異なる．①前者の場合には，端のストロークが残る．②後者の場合には，筆順の上で離れているものが残る．

①の例を図 7-(d)に示す．その割当を表2のd欄に示す．この場合には，未割当の実参照ストロークの前あるいは後ろの参照ストロークに割り当てられている入力ストロークはこの実参照ストロークまで含むものと見なせばよい．いずれの側に含ませるかは類似度の高いほうにする．

②の例を図 7-(e)に示す．その割当は表2のe欄のようになる．これは「ふるとり」の中の連続していない斜め棒と縦棒とを続けて書いたものである．ヒューリスティックにストローク合わせを行う方法では，このような参照ストロークを作り出すことができないので，本質的な解決をすることはできない．しかし，この形態については，その数は余り多くないので，次の方法で二義的に解決する．未割当の実参照ストロークを隣の参照ストロークと結合し，それに割り当てられている入力ストロークに対応するものと便宜的に見なす．先の例に対しては，一番上の横棒の入力ストロークに対して，斜め棒と横棒を結合した参照ストロークで対応させる．

これまでは入力パターンと参照パターンとが同一の漢字である場合を想定して議論を進めてきたが，認識の過程ではさまざまな参照パターンについてこれまでの操作を行う．そこで，余り似ていない参照パターンに対しては仮説の設定が生じないように次の処理を行う．だぶり解消の手順でだぶりが解消できないとき，あるいは，未割当解消の操作である入力ストロークに対してある程度以上の類似度を有する参照ストロークが割り当てられないときは，参照パターンは入力パターンに対応しないものとし，次の参照パターンの処理に移る．

5. 評　　価

ヒューリスティックなストローク合わせと仮説設定による方法での手書き漢字に対する認識実験を次の要領で行った．参照パターンとして教育漢字 996 文字をすべて登録した．実験の対象にした書体は楷書体と行書体で，5, 9, 14 画の教育漢字 70, 102, 52 字すべてを対象にした．被験者の数は，楷書体のとき 10 人，行書体のとき 16 人である．楷書体の漢字については，正しい画数で書くという制約をもうけたが，書き順については制約をもうけていない．行書体の漢字については，文献 13) を参考に入力してもらった．なお，結合してもよい箇所は入力パターンあたり7を最大とし，また，1つのストロークでの結合箇所は最大3とした．したがって，入力パターンのストローク数が7画であるときは7画から 14 画までの参照パターンとの間で認識を行うこととなる．**表 3** に実験結果を示す．これには，画数ごとの認識結果と，それらの合計を示した．また，正しい参照パターンが出現した順位を累計で示した．**図 8** に評価に使った行書体の漢字の例を，また，**表 4** にその崩され具合を示す．

ヒューリスティックにストローク合わせを行う方法では，必ずしも正しいストローク合わせになるとは限らない．そこで，この方法の有効性を示すためには全数探索によるストローク合わせの方法との間にどれだけの差があるかを示す必要がある．**表 5** には動的計画法でストローク合わせを行ったときの認識結果を示

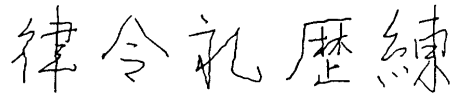


図 8 評価に使われた行書体の入力パターンの例
Fig. 8 Input pattern samples for evaluation.

表 3 実験結果
Table 3 Evaluation results.

	5画 (70字)			9画 (102字)			14画 (52字)			合　計		
順位	1位	3位	5位	1位	3位	5位	1位	3位	5位	1位	3位	5位
楷書体	95.0	99.3	99.7	95.4	98.6	99.2	95.2	99.8	99.8	95.2	99.1	99.5
行書体	71.3	88.9	92.9	75.7	90.0	93.4	81.2	91.7	94.7	75.6	90.1	93.6

表 4 行書体漢字の崩しの度合い
Table 4 The cursiveness degree for input patterns.

	楷書体とのストローク差（結合数）								平均ストローク差
	0	1	2	3	4	5	6	7	
5 画	10.0	50.0	32.9	7.1					1.4
9 画	2.9	25.7	47.1	47.1	18.6	4.3			2.5
14 画		5.8	9.6	21.2	28.8	25.0	5.8	3.8	3.9

表 5 ヒューリスティックなストローク合わせ法 (HSM) と動的計画法 (DP) の比較
Table 5 The recognition rate comparison between heuristic stroke matching and dynamic programming.

画数		5画			9画			14画		
順位		1位	3位	5位	1位	3位	5位	1位	3位	5位
楷書体		97.4	99.9	100.0	99.4	100.0	100.0	99.8	100.0	100.0
行書体	HSM	78.6	88.6	92.9	88.2	95.1	97.1	94.2	98.1	98.1
	DP	80.0	90.0	97.1	89.2	98.0	98.0	96.2	98.1	100.0

表 6 処理時間の比較
Table 6 Processing time for on-line recognition.

機種	CPU	クロック	楷書体	行書体
9801 vm	V30	10 MHz	16.3秒	268.4秒
98 RL	80386	20 MHz	2.2秒	35.3秒
NEWS 1960	68030	25 MHz	1.2秒	20.0秒

す．楷書体の漢字については，先の評価のときに用いたすべての入力パターンについて評価を行ったが，行書体の漢字については，相当な計算時間を必要とするため，参照パターンを書いた被験者の入力パターンについてのみ行った．このため，これについては，ヒューリスティックなストローク合わせのときの認識結果もあわせて記した．これによれば，その差は数%である．後で述べるように処理時間が大幅に短縮されたことを考えれば有効な方法であるといえる．

図 7-(b)と(e)の楷書体と行書体での入力パターン「雑」に対する処理時間を**表 6** に示す．「雑」の行書体は 8 画であるため 8 画から 15 画までの参照パターンとの間で認識を行ったことになる．この画数の範囲内の教育漢字の数は 635 である．したがって，楷書体のときと比較すると参照パターンの総数は 12 倍である．処理時間の比もそれに近いことが分かる．これらの入力パターンに対して，論文[4]の動的計画法でストローク合わせを行ったときの処理時間は楷書体に対しては 7.8 秒，行書体に対しては 41 分であった．これから，ヒューリスティックにストロークを合わせる方法はきわめて有効であるといえる．

6. おわりに

楷書体から行書体までの手書き漢字をファジィ推論を用いてオンラインで認識する方式を提案し，その実験結果を示した．続け字と崩し字に対応できるようにするために，楷書体で与えられた参照パターンから，隣合うストロークを続けることによって続け字および崩し字を自動的に生成する方法を提案した．また，漢字の特徴を表すために，各ストロークを周波数領域で表すとともに，それが持つ曖昧性を表現するためにそれをファジィ化データで表す方法について提案した．さらに，参照パターンについてはファジィ化データからプロダクションルールを自動的に生成する方法を，入力パターンについてはプロダクションルールからファジィ推論により認識する方法を提案した．また，プロダクションルールの適用が全数探索になることを防ぐために，ヒューリスティックにストローク合わせを行い，ストロークの対応関係に対する仮説を設定して，利用するプロダクションルールの数を大きく削減する方法についても述べた．

実験結果によれば，正しく認識した割合は楷書体で 95.2%，行書体で 75.6% であった．また，5 位までで正しく認識した割合はそれぞれ 99.5%，93.6% であった．また，認識時間は 32 ビットマイクロプロセッサ 80386 (20 MHz) で，「雑」の楷書体に対して 2.2 秒，行書体に対して 35.3 秒であった．楷書体に対しては 32 ビットのマイクロプロセッサを用いれば十分であると考えられるが，行書体に対しては実用的

なレベルに持っていくためには，さらに 10 倍程度高速にする必要がある．これに対しては，専用のハードウェアチップを開発することが1つの解決策と考えられる．また，認識率をさらに向上させるためには，偏や旁を考慮した方法も有力と考えられる．さらに，ここで述べた方法はストロークの抽出が可能になればオフラインの手書き漢字認識にも応用することができると考えられる．したがって，これらのテーマについて今後も研究を継続していきたい．

参考文献

1) 小高和己，若原　徹，増田　功：筆順に依存しないオンライン手書き文字認識アルゴリズム，信学論 (D), Vol. J 65-D, No. 6, pp. 679-686 (1982).
2) 萬木正義，永田静男，小沼賢二，久保田恵子：階層分析法によるオンライン文字認識，信学論 (D), Vol. J 68-D, No. 6, pp. 1320-1327 (1985).
3) 石井康雄：ストローク代表点に着目したオンライン手書き漢字認識，信学論 (D), Vol. J 69-D, No. 6, pp. 940-948 (1986).
4) 大森健児：ファジー推論による実時間手書き漢字認識，信学論 (D), Vol. J 72-DII, No. 3, pp. 369-379 (1989).
5) 沢井良一，中川正樹，高橋延匡：オンライン手書き日本語文字認識におけるつづけ字の検討，信学論 (D), Vol. J 70-D, No. 5, pp. 946-957 (1987).
6) 佐藤幸雄，足立秀鋼：走り書き文字のオンライン認識，信学論 (D), Vol. J 68-D, No. 12, pp. 2116-2122 (1985).
7) 若原　徹，梅田三千雄：ストローク結合規則を用いたオンラインくずし字分類，信学論 (D), Vol. J 67-D, No. 11, pp. 1285-1292 (1984).
8) 若原　徹：局所的 Affine 変換を用いたオンライン手書き文字認識，信学論 (D), Vol. J 71-D, No. 2, pp. 379-386 (1988).
9) 蔭田千冬，中川正樹，高橋延匡：オンライン手書き文字認識における，サブパターンの導入による略字，くずし字，筆順誤りへの対応，信学論 (D), Vol. J 70-D, No. 4, pp. 777-784 (1987).
10) Granlund, G. H.: Fourier Pre-processing for Hand Print Character Recognition, *IEEE Trans. Comput.*, Vol. C-21, No. 2, pp. 195-201 (1972).
11) Persoon, E. and Fu, K.: Shape Discrimination Using Fourier Descriptors, *IEEE Trans. Syst. Man Cybern.*, Vol. SMC-7, No. 3, pp. 170-179 (1977).
12) 上坂吉則：閉曲線にも適応できる新しいフーリエ記述子，信学論 (A), Vol. J 67-A, No. 3, pp. 166-173 (1984).
13) 下村芳流：ペン字三体手本，永岡書店 (1986).

(平成元年8月26日受付)
(平成2年3月 6 日採録)

大森　健児 (正会員)

昭和 44 年東京大学工学部計数卒業．昭和 47 年カリフォルニア大学バークレイ校大学院修士課程修了．昭和 44 年日本電気入社．昭和 60 年法政大学教授．マルチプロセッサシステム，CAD 専用装置，オブジェクト指向言語，オンライン手書き漢字認識，金型設計エキスパートシステムなどの研究に従事．工学博士．昭和 59 年情報処理学会論文賞，情報処理学会 25 周年記念論文に選定．著書「計算機アーキテクチャ」(オーム社，共著) など．電子情報通信学会，日本ソフトウェア科学会，AAAA, IEEE Computer Society 各会員．