

論文

楷書オンライン入力からの連筆文字生成法について

山田 晃嗣¹⁾ 江野脇 宏²⁾ 中村 剛士¹⁾ 何 立風³⁾ 伊藤 英則¹⁾

筆者らは、これまでユーザの個性を反映させた楷書毛筆文字の生成に関する研究を行ない、いくつかのシステムを提案・構築してきた。しかしながら、それらのシステムでは、楷書及び平仮名以外の毛筆書体を扱うことができないという課題が残っていた。そこで本稿では、連筆毛筆文字生成システムを提案・構築する。提案手法では、ユーザが電子ペンで入力した楷書文字を連筆文字へ変換するという手法を用いているため、同一入力データから楷書文字と連筆文字双方のスタイルを同時に生成・獲得できるという特徴がある。また、このシステムを用いることで、ユーザは連筆文字の描画に関する知識がなくても、容易に連筆文字を作成できる。

キーワード：連筆文字、ファジィ・ニューラルネットワーク、行列変換

1. はじめに

近年、パーソナルコンピュータの普及によりその利用形態が多様化したことから、様々なソフトウェアが開発・提供されている。そうした中で、アプリケーション・ソフトウェアの高機能化、周辺装置の高性能化に伴い、出力文字(フォント)の多様化・高品質化が望まれている。日本においては、フォントを使用する主たるソフトウェアの1つとして、年賀状作成などの用途で用いられる葉書印刷ソフトウェアを挙げることができる。この種のソフトウェアのほとんどにおいては、楷書体、行書体、相撲文字など多種類の毛筆フォントを提供し、さらに一部のソフトウェアでは、文字間につながりがある連綿体や掠れや滲みのある毛筆フォントなども登場してきている。これは、ユーザからのフォント品質に対する要望の高まりを反映したものと考えができる。しかし、フォントの品質が高まる一方で、ユーザの個性(筆記技能や癖など)まで反映した各ユーザ独自のオリジナル・フォントを提供するシステムやサービスはまだ発展途上の段階である。

我々は、この点に着目し、マウスや電子ペンによっ

て入力された座標データ、筆圧、筆速などの情報を処理して、ユーザ個人のオリジナル毛筆フォントを作成するシステムをこれまで研究・開発してきた[1, 2, 3, 4, 5]。これらのシステムは、マウスや電子ペンを入力装置として用いているため、個人の筆記技能が文字形状に反映される。我々の従来研究において対象としてきた毛筆文字は楷書漢字と平仮名のみであるが、専門的書字技能を有する者であれば、行書や草書などの連筆文字についても、従来システムを使用することによって作成可能である。しかしながら、一般的には楷書についての書道教育を受けた人口と行書や草書のような高度書道技術について教育を受けた人口を比較した場合、後者が圧倒的に少ないとと思われることから、行書や草書にたいする従来システムの有用性は低い。また、行書や草書はその作成ルールが複雑であり、これまで我々が研究・開発してきたシステムを用いることによって、楷書教育しか受けていない一般の人が行書や草書を作成することは困難に近い。このことから、実用性の点で大きな課題があるといえる。

そこで本稿では、連筆文字を作成する手法として、オンライン入力した楷書骨格から連筆文字を作成するシステムを提案する。本システムの特徴としては、まず(1)従来システム同様、入力装置に電子ペンを用いており、これを用いることによって、出力毛筆文字にユーザの個性を反映させることができる。また、(2)オンライン入力された楷書骨格を連筆文字に自動変換するため、楷書教育しか受けてないユーザであっても連筆文字を容易に作成することが可能である。すなわち、ユーザに要求する知識や操作は従来システムと同程度でありながらも、本システムは連筆文字への対応／拡張を実現する。なお、この楷書骨格から連筆骨格への

Transforming Technique from on-line inputted *Kai-sho* style to *Ren-hitsu* style

Koji YAMADA, Hiroshi ENOWAKI, Tsuyoshi NAKAMURA, Lifeng HE and Hidenori ITOH

¹⁾ 名古屋工業大学 知能情報システム学科

Dpt. of Artificial Intelligence and Computer Science,
Nagoya Institute of Technology

²⁾ 日立ソフトウェアエンジニアリング

Hitachi Software Engineering Co., Ltd.

³⁾ 愛知県立大学情報科学部地域情報科学科

Faculty of Information Science and Technology, Aichi
Prefectural University

変換については、ファジィ・ニューラルネットワーク(以下、FNN)[6, 7, 8, 9]によってストローク間の連結基準を学習し、そこで得られた連結基準ファジィール用いてストローク間を連結するか否かを決定する。さらに、楷書骨格から連筆骨格への変形処理は行列変換によって実現する。

2. システムの概要

本システムの構成について述べる。本システムは、図1に示すように、入力部、連筆変換部、毛筆文字変換部、出力部の4つのパートから構成される。なお、連筆変換部については、前処理部、FNNおよび文字変形部から構成される。処理のながれとしては、まず、電子ペンによりユーザが楷書骨格をオンライン描画入力する。つぎに、入力された楷書骨格データを連筆スタイルに変換し、さらにこれにたいして毛筆文字化処理を施すことで、連筆文字を作成する。以下に各部において実行する処理の概要を述べる。

入力部：ディスプレイ画面上に表示されたウィンドウ上でカーソルを動かすことによって、ユーザは楷書骨格を描画入力する。入力には、電子ペンおよび専用タブレットを用いる^{*1)}。電子ペンによってウィンドウ上に描画された楷書骨格から、入力ストロークデータ(各サンプリング座標、サンプリング時刻、筆圧)を獲得する。

連筆変換部：まず、前処理部では、入力ストロークデータをキーとして分類データベース^{*2)}を検索し、入力ストロークに対応する分類データベース上の標準ストロークを特定する。この認識結果をもとに、入力ストロークをセグメント^{*3)}に分割し、セグメント情報を抽出する[1]。つぎに、そのセグメント情報とあらかじめ学習により獲得した連結基準をもとに、FNNが楷書骨格を構成するストローク同士を連結するか否かの程度(連結度)を決定・出力する。ここでは、この連結度が一定値以上の場合は、ストロークを連結し、変換行列を用いて楷書骨格から連筆骨格への変形を実行する。なお、連筆変換部の処理については次節以降で詳細を述べる。

脚注^{*1)} このとき、描画入力は楷書ストロークごとに行なうため、1ストロークを描画完了するごとに、タブレット上から電子ペンを離す操作をユーザは行なう必要がある。

脚注^{*2)} 楷書漢字を構成する46種類のストロークを格納したデータベースであり、文献[1, 2, 4]中で使用している。

脚注^{*3)} セグメントとは、ストロークの始筆点から終筆点までを曲折点において分割したときに得られる線分である。

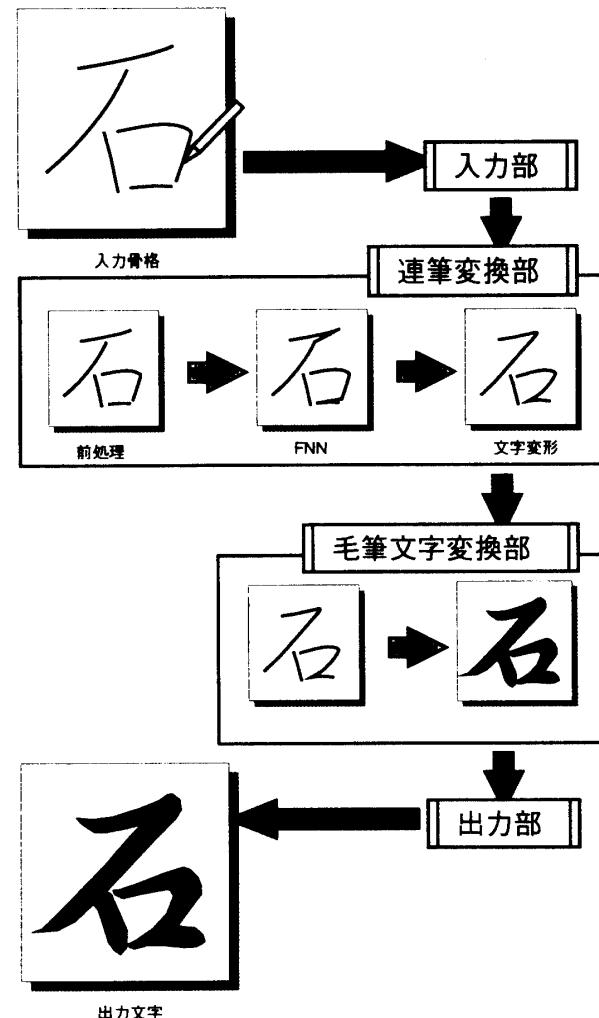


図1. システムの概要

毛筆文字変換部：連筆骨格にたいして肉づけを行ない、毛筆文字スタイルに変換して連筆文字を作成する。なお、この毛筆文字化処理には、運筆状態遷移規則[5]を用いる。

出力部：連筆文字に掠れ・滲み処理[3]を施し、画面上に出力・表示する。

3. 連筆文字変換

連筆文字とは、楷書文字の連続するストローク間、すなわち第*i*ストロークと第*i+1*ストローク間を繋いで描画するいわゆる“つづけ字”である。この連続するストローク間を繋ぐ際には、むやみやたらと繋げるのではなく、多くの人が行なってきた普遍性から生まれた美的基準が存在する[10]。

この基準は、繋げるべきストロークの種類、連続するストロークの相対位置などから求めることができると考えられるが、書道専門書などにはその明確な定義は示されておらず、事例によって解説されている場合が多い。

そこで本稿では、連続する2ストローク間の連結についてFNNを用いて学習を行ない、連筆基準についてのファジィ推論ルールを獲得する。FNNはその内部構造の特性から、学習によって獲得されたルールがそのままファジィ推論の形式で表されることから、ルールが直観的で理解しやすい。また、ルールの統合およびチューニングによっては高速化も期待できることから、ここでは、FNNを用いて連筆基準の獲得を試みた。

楷書のストロークは、46種類に分類することができるが[1]、この場合、連続する第*i*ストロークと第*i+1*ストロークの考えうる組み合わせは $46 \times 46 = 2116$ 通りにもなる。そこで、ここでは、ストロークをセグメントに分割し、第*i*ストロークを構成する最後のセグメントと第*i+1*ストロークを構成する最初のセグメントの組み合わせを考えるものとする。本稿では、セグメントを(1)縦棒、(2)横棒、(3)左払い、(4)右払い、(5)右はね、(6)点画の6種類に大分類し、組み合わせ数を $6 \times 6 = 36$ に限定してFNNで扱うこととする。

なお、FNNの学習データ作成には、行書専門書の1つである文献[10]に記載の行書文字例とそれに対応する楷書文字について、書道専門家に電子ペンによって描画入力してもらいデータ化した。すなわち、その入力楷書文字データから連続するストロークを抽出し、セグメントに分割した後、第*i*ストロークを構成する最後のセグメントと第*i+1*ストロークを構成する最初のセグメントの関係を教師入力とした。教師出力としては、連続するストロークを連筆文字として連結すべき場合は0.99、連結すべきでない場合は0.01を与えて学習を実行した。

3.1 前処理部

前処理のながれは以下のとおりである。

1. 分類データベースを検索し、各入力ストロークの形状認識を行ない、入力ストロークに対応する標準ストロークを特定する。ここで、第*i*画目の入力ストロークを S_i として表すこととする($0 \leq i \leq N-1$)。なお、 N は1文字中の総入力ストローク数である。
2. 検索結果から得られた標準ストローク情報から、 S_i をセグメントに分割する。なお、 S_i の第*j*番目のセグメントは $s_{i,j}$ として表す($0 \leq j \leq m_i-1$)。なお、 m_i は第*i*ストロークを構成する総セグメント数であり、入力ストロークが対応する標準ストロークによって一意に定まる。
3. 各入力ストローク S_i の第0セグメント $s_{i,0}$ と第 m_i-1 セグメント s_{i,m_i-1} を直線近似する。

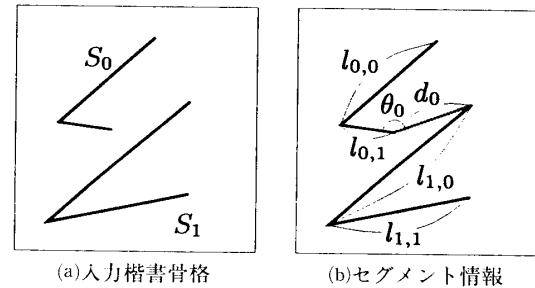


図2. 抽出するセグメント情報の例

4. 直線近似したセグメントから、セグメント情報を抽出する。なお、セグメント情報とは、セグメント長、ストローク間距離、ストローク間のなす角を示し、各定義は次のとおりである。まず、ここで用いるセグメント長とは、各入力ストロークの第0セグメント $s_{i,0}$ と第 m_i-1 セグメント s_{i,m_i-1} の長さ $l_{i,0}$ と l_{i,m_i-1} を表す。また、ストローク間距離とは、 s_{i,m_i-1} の終筆点と $s_{i+1,0}$ の始筆点間のユークリッド距離 d_i を表す。さらに、ストローク間のなす角とは、 s_{i,m_i-1} の終筆点と $s_{i+1,0}$ の始筆点間を結ぶ直線と直線近似した s_{i,m_i-1} のなす角 θ_i を表すものとする。抽出するセグメント情報の例として、“糸”的第0画と第1画について図2に示す。

3.2 ファジィ・ニューラルネットワーク

本稿で用いるFNNは、図3に示す構成になる。ここでは、第*i*ストロークの最後のセグメントと第*i+1*ストロークの最初のセグメントの組み合わせ数、すなわち36個のFNNを用意し、両セグメントの組み合わせによって、使用するFNNを唯一決定する。

このFNNは、以下のファジィ推論ルールをネットワークによって表現したものであり、その入出力関係を表1に示す。前件部パラメータについては、前処理によって獲得したセグメント情報である。また、後件部出力値はそれらから推論される連結度であり、0.0~1.0の範囲で値をとり、値が大きいほど連結度が強いものとする。式(1)中の A_j ($j=1, \dots, 5$)は、入力変数 a_j の前件部メンバーシップ関数であり、図3中の K_j ($j=1, \dots, 5$)はすべて3である。

$$\begin{aligned} \text{if } & a_1 = A_1 \text{ and } a_2 = A_2 \text{ and } a_3 = A_3 \\ & \text{and } a_4 = A_4 \text{ and } a_5 = A_5 \\ \text{then } & \gamma_i = B \end{aligned} \quad (1)$$

なお、FNNの第2中間層から出力層間のリンク荷重は、事前に最急降下法によりそれぞれのFNNにおいて学習を実行して準最適な荷重を設定するものとする。

表1. FNNの入出力変数

入力変数	a_1	S_{i,m_i-1} の長さ l_{i,m_i-1}
	a_2	$S_{i+1,0}$ の長さ $l_{i+1,0}$
	a_3	$S_{i,m_i-1}, S_{i+1,0}$ 間の距離 d_i
	a_4	$S_{i,m_i-1}, S_{i+1,0}$ のなす角 θ_i の余弦 $\cos \theta_i$
	a_5	$S_{i,m_i-1}, S_{i+1,0}$ のなす角 θ_i の正弦 $\sin \theta_i$
出力変数	γ_i	S_i, S_{i+1} 間の連結度

表2は、2つの横棒セグメントの組み合わせについて獲得した連筆基準ファジィルールの一部である。なお、とくに後件部出力が1.0に近いものを示した。獲得ルール数が膨大なためここですべてを示すことはできないが、出力値 γ_i が $\eta_0^{*4)}$ 以上をとる各FNNの平均ルール数は52個であった。また、獲得されたルールからは次のような傾向が見られた。

まず、セグメントの組合せによるルールの差異があまりなく、構成した36個のFNNが比較的類似していることが挙げられる。これはすなわち、ルールがセグメントの持つ固有の形状にのみ依存するのではなく、相対的位置関係への依存性が高いと考えることができる。これについては、表2が一部示すように、セグメント長を表す入力変数 a_1, a_2 へのルールの依存が比較的見られないことからも分かる。また、ストローク間距離 a_3 を示す入力変数 a_3 が“短い”となるルールの数は、1つのFNNの総ルール数243にたいし81個存在するが、すべてのFNNでの総ルールでは平均して33個のルールの出力値 γ_i が η_0 以上であり、連結すべきと判断されることからも相対位置関係への依存性の高さが伺え、ストローク間距離が重要なパラメータとなっていることが分かる。さらに、セグメントのなす角の余弦 $\cos \theta_i$ を表す a_4 が大きい場合に連結度に強い影響を与えることも獲得されたルールから分かっている。これは、第 $i+1$ ストロークが第 i ストロークの終筆部分における進行方向と逆方向に位置することを意味し、第 i ストロークを終筆部分で止めた後、第 $i+1$ ストロークに向かって払う動作によって連結が成立する場合が少なからず存在すると考えることができる。

3.3 文字変形部

文字変形部では、楷書骨格から連筆骨格への変形処理を行なう。文献[11]は楷書体の各ストロークを直線補間したものから崩し漢字を生成する手法として、行列変換を用いる手法を提案している。この手法では、次に描かれるであろう先々の描画曲線を現時点に取り込む処理に生成行列を変換行列として用い、単純な補

脚注^{*4)} ここでは、 η_0 を0.70として設定し、FNNがそれ以上の値を出力した場合、連結処理を実行する。図4参照。

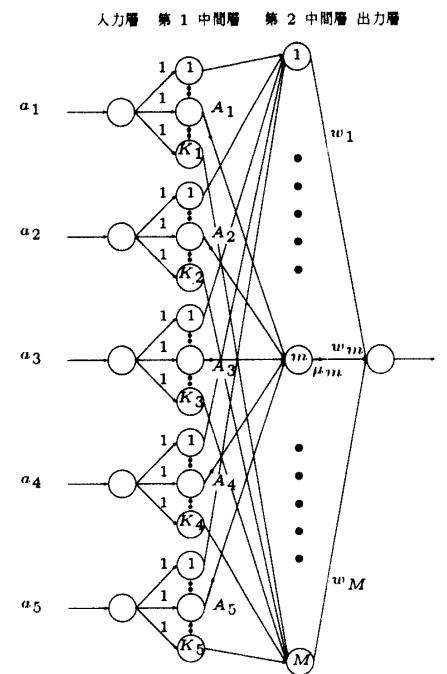


図3. ファジィ・ニューラルネットワーク

表2. 獲得された連筆基準ファジィルールの一部

ルール番号	前件部					後件部
	a_1	a_2	a_3	a_4	a_5	
1	短い	短い	短い	大きい	普通	0.99
2	短い	短い	短い	普通	大きい	0.93
3	短い	短い	短い	小さい	普通	0.98
4	短い	普通	短い	大きい	普通	0.90
5	普通	普通	短い	普通	大きい	0.89
6	普通	短い	短い	小さい	普通	0.97
7	普通	普通	普通	大きい	大きい	0.89
8	普通	短い	普通	大きい	大きい	0.99
9	短い	普通	長い	大きい	大きい	0.98
10	短い	長い	普通	大きい	普通	0.98
11	普通	長い	長い	大きい	普通	0.95
12	長い	長い	長い	大きい	普通	0.96

間法や関数近似と比較して、より実際に近い崩し漢字の生成を実現できるとしている。そこで、本稿では、この手法を応用し、楷書骨格から連筆骨格への変換を行行列変換によって行なう。

3.3.1 変形処理アルゴリズム

図4に、楷書骨格から連筆骨格への変形処理アルゴリズムを示す。また、図5(a)～(f)に本アルゴリズムによる連筆変形処理過程のイメージ図を示す。以下、図5を例として図4に示したアルゴリズムについて述べる。

図5(a)の処理対象である楷書文字“石”は、入力楷書ストローク総数 $N=5$ であり、入力楷書ストローク S_i 、オフストローク O_i 、連結度 γ_i ($i=0, 1, \dots, N-1$) か

```

1 begin
2    $i := 0, j := 0, p := 0, \Gamma := 0;$ 
3   repeat
4      $T_j := S_i;$ 
5     while( $\gamma_i > \eta_o$ )
6       begin
7          $\Gamma := \Gamma + \gamma_i;$ 
8          $i := i + 1;$ 
9          $p := p + 1;$ 
10         $T_j := append(T_j, O_{i-1}, S_i);$ 
11      end
12     $m := f(\Gamma, p);$ 
13     $T'_j := trans(T_j);$ 
14     $i := i + 1, j := j + 1, p := 0, \Gamma := 0;$ 
15  until( $i < N$ )
16 end

```

図4. 連筆文字変換アルゴリズム

ら構成される。ただし、ここでは、 O_{N-1} および γ_{N-1} は考えないものとする。図5(b)にオフストロークとそれぞれの連結度を示す。

図4の6行目～12行目は、 γ_i が η_o より大きい場合に入力楷書ストロークを $append$ 関数によって順次連結することを意味する。これにより、 S_o, O_o, S_i から図5(c)に示す連結楷書ストローク T_o が生成される。さらに、 T_o は取り込み数 m にしたがって、図4の14行目 $trans$ 関数によって連筆ストローク T'_o に変形される。さらに、同様の処理によって S_2, O_2, S_3, O_3, S_4 から T_1, T'_1 が生成される(図5(d)(e)参照)。なお、 m および $trans$ 関数については次節で詳述する。

3.3.2 変換行列

$trans$ 関数は、連結楷書ストローク T_j から、連筆ストローク T'_j を生成する関数である。以下、 $trans$ 関数について述べる。まず、 T_j を構成する座標点列は、総座標点数を n_{T_j} として式(2)のように記述する。なお、 t_k^i は、 k 番目の座標点を複素数表現したものである。 T_j は隣接座標点間のベクトルを表す W に置き換え、文献[11]において提案されている変換行列 $[C_m]$ を利用して W' に変換する。連筆ストローク T' は、 W' を座標点列に逆変換することで獲得される。

$$T_j = (t_0^i, t_1^i, \dots, t_k^i, \dots, t_{n_{T_j}-1}^i) \quad (2)$$

$$\begin{aligned} W &= (w_0^i, \dots, w_k^i, \dots, w_{n_{T_j}-2}^i) \\ &= (t_1^i - t_0^i, \dots, t_{k+1}^i - t_{n_{T_j}-1}^i - t_{n_{T_j}-2}^i) \end{aligned} \quad (3)$$

変換行列 $[C_m]$ は、式(4)に示すように、 $[A_m]$ と $[B_m]$ の和で構成される。 $[A_m]$ と $[B_m]$ については、それぞれ $[\alpha]$ と $[\beta]$ のべき乗和で構成される。式(7), (8)に示す

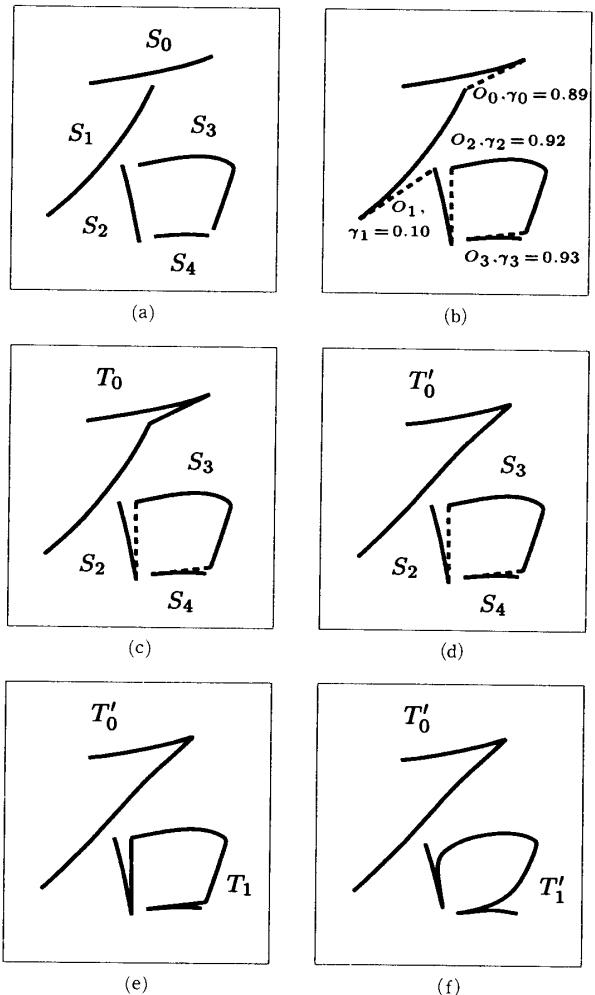


図5. 楷書骨格から連筆骨格への変換

ように、 $[\alpha]^k$ は曲線のベクトル列を上へ k 個ずらす作用をし、また $[\beta]^k$ は、曲線のベクトル列を下に k 個ずらす作用をする。したがって、 $[A_m], [B_m]$ によって変換される曲線は、ベクトル列がずれたものの和として表現できる。図6に、 $[A_m], [B_m]$ によって変換される簡単な曲線の例を示す[11]。図6に示す例は、ベクトル数 $n=3$ 、自分自身のベクトルを含めた取り込み数 $m=2$ の場合の変換の例である。 $[C_m]$ については、これら基本生成行列 $[A_m], [B_m]$ を基にして定義した連結楷書ストロークから連筆ストロークへの変換行列であり、自分自身のベクトルと前後 $m-1$ 個のベクトルの和を計算し、新しいベクトルを生成する。なお、本稿では文字サイズを大きく変更しないために、生成された各ベクトルを $1/(2m-1)$ している(図4参照)。

$$[C_m] = [A_m] + [B_m] - [I] \quad (4)$$

$$[A_m] = [I] + [\alpha] + [\alpha]^2 + \dots + [\alpha]^{m-1} \quad (5)$$

$$[B_m] = [I] + [\beta] + [\beta]^2 + \cdots + [\beta]^{m-1}$$

$$[\alpha] = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & 1 \\ 0 & \cdots & 0 & 0 & 0 \end{bmatrix} \quad (6)$$

$$[\beta] = \begin{bmatrix} 0 & & \cdots & 0 \\ 1 & 0 & & \cdots & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & \cdots & 1 & 0 & 0 \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} w_k \\ w_{k+1} \\ \vdots \\ w_{n-1} \\ 0 \\ \vdots \\ 0 \end{bmatrix} = [\alpha]^k \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_k \\ w_{k+1} \\ \vdots \\ w_{n-1} \end{bmatrix} \quad (7)$$

$$\begin{bmatrix} 0 \\ \vdots \\ 0 \\ w_0 \\ w_1 \\ \vdots \\ w_{(n-1)-k} \end{bmatrix} = [\beta]^k \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_{(n-1)-k} \\ w_{(n-1)-k+1} \\ \vdots \\ w_{n-1} \end{bmatrix} \quad (8)$$

$$W' = [A_m] W =$$

$$\begin{bmatrix} w_0 \\ \vdots \\ \vdots \\ \vdots \\ w_{n-1} \end{bmatrix} + \begin{bmatrix} w_1 \\ \vdots \\ \vdots \\ w_{n-1} \\ 0 \end{bmatrix} + \cdots + \begin{bmatrix} w_{m-1} \\ w_m \\ \vdots \\ 0 \\ \vdots \end{bmatrix} \quad (9)$$

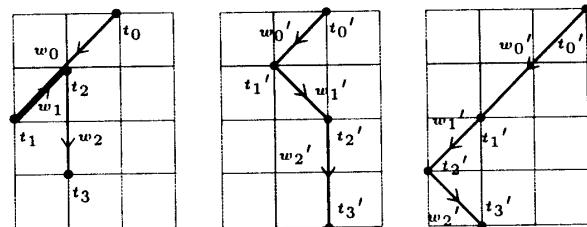


図 6. $[A_2]$, $[B_2]$ による曲線の変換

$$W' = [B_m] W =$$

$$\begin{bmatrix} w_0 \\ \vdots \\ \vdots \\ w_{n-1} \end{bmatrix} + \begin{bmatrix} 0 \\ w_0 \\ \vdots \\ \vdots \\ w_{n-2} \end{bmatrix} + \cdots + \begin{bmatrix} 0 \\ \vdots \\ 0 \\ w_0 \\ \vdots \\ w_{n-1-(m-1)} \end{bmatrix} \quad (10)$$

4. 連筆文字生成

図 7 に、本提案手法によって作成した連筆文字の出力例を示す。図 7(a)は入力楷書骨格であり、各ストローク間を繋ぐ破線は、FNN がそのストローク間の連結を決定したことを意味するまた、変形処理アルゴリズム中の関数 $f(\Gamma, p)$ を $f_1(\Gamma, p)$, $f_2(\Gamma, p)$ とした場合に出力される連筆文字例をそれぞれ図 7(b)および(c)に示す。さらに、図 7(b)の連筆骨格を毛筆文字化処理によって毛筆文字に変換し、掠れ・滲み処理[3]を実行した例を図 8 に示す。

$$f_1(\Gamma, p) = \begin{cases} 10\Gamma/p + 2 & (p \neq 0) \\ 2 & (p=0) \end{cases} \quad (11)$$

$$f_2(\Gamma, p) = 2f_1(\Gamma, p) \quad (12)$$

ここで、特に高度な書字知識をもたない被験者30名を用意し、本システムによって作成した図 7 と他の連筆文字 8 文字を加えた合計12文字にたいする主観評価を実施した。評価項目と評価結果を表 3～5 に示す。項目内の数字は各評価をつけた人数を示す。表 3 は、連結したストロークの選択・決定が適切であったか否かの評価に相当し、評価が高ければ、FNN の学習が適切であったといえる。表 4 は、生成した連筆文字の形状についての評価であり、行列変換が適切に作用したか否かを示す。表 5 については、このシステムの有用性評価の結果である。また、書道専門家の主観評価として、次のような意見があった。



図7. 連筆骨格の作成例

表3. 主観評価1

	適切	ほぼ適切	やや不適切	不適切
連筆箇所	15	14	1	0

表4. 主観評価2

	自然	ほぼ自然	やや不自然	不自然
連筆形状	4	13	9	4

表5. 主観評価3

	非常に高い	高い	並	低い	非常に低い
有用性	2	11	7	9	1

- 連筆されている箇所はほぼ適切であるが、図7中の“春”的6-7画間や、“夏”的3-4画間は連結されるべきである。
- 文字のくずし方に不自然な箇所が所々に見受けられる。たとえば、図7 “春”の“日”部分の2画目などである。
- 文字のくずしが軟らかすぎる傾向がある。もっと筆の勢いを残した方がいい。
- 全体的に連筆文字らしい表現を実現してはいるが、現段階では連筆文字に類似した表現の域である。

これらの評価を総合すると、連結すべきストロークの選択には一部問題があるが、全体的にはほぼ適切であるといえる。一方、連筆文字への変形については、文字の品質を損なうほど視覚的に耐え難いわけではないが、実際に専門家が筆を用いて作成した連筆文字と比較した場合や芸術的表現の観点から見た場合、不備があると言わざるを得ない。ただし、システムの有用性については被験者30名中2/3近くが「並」以上の評価を与えており、高度な書字知識を持たない人には、ある程度有効なシステムではないかと思われる。

5. おわりに

本研究では、連筆文字を作成する手法として、オンライン入力した楷書骨格から連筆文字へ変換するシステムを提案し、それをWS上に実装した。本提案システムは、従来システム同様、入力装置に電子ペンを用いており、これによって出力される毛筆文字にユーザの個性を反映させることができる。また、オンライン入力された楷書骨格を連筆文字に変換処理するため、楷書教育しか受けていないユーザであっても容易に連筆文字を作成することが可能である。また、本提案手法に類似した処理としてとしては、文字間の連結いわゆる連綿体を生成するパッケージソフトが存在するこ

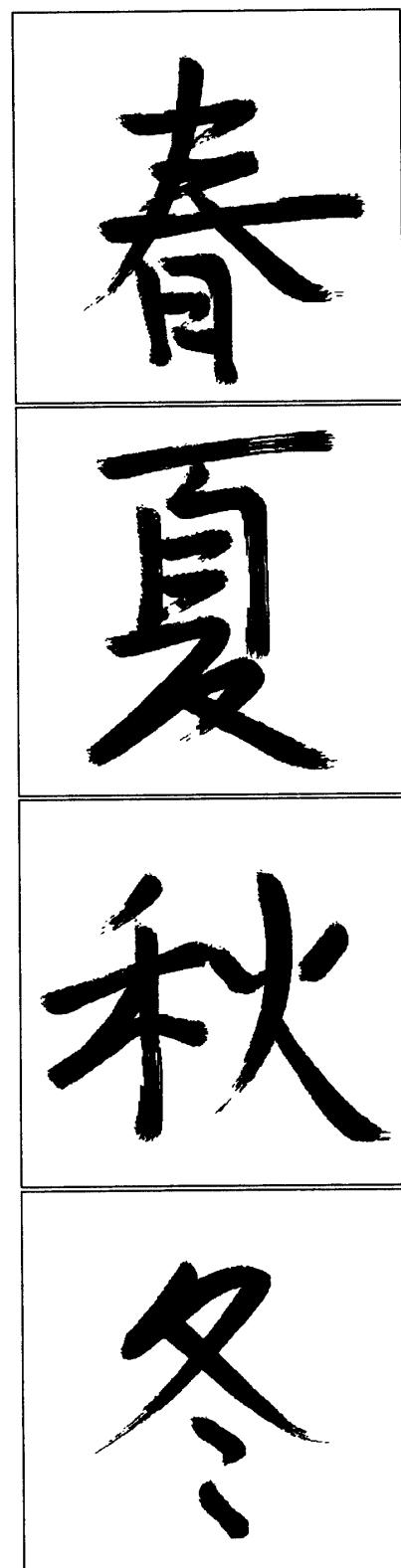


図8. 連筆文字の作成例

とを唯一確認しているのみであり、楷書から連筆文字を生成する類似アプローチについては現時点では確認されていない。連綿体生成も書道技法の1つであり重要な地位を占めるが、本提案手法のように、文字自体の形状を変形するまでの処理は行っておらず、本研究の目指すユーザ自身の個性を反映した連筆文字生成と

は異なる処理である。

今回、ストローク間の連結基準をFNNを用いて獲得したが、本システムの出力例を見る限り、学習後のFNNがある程度機能していることが分かることから、これについては一定の評価を与えることができる。しかし、学習事例数が多くないことから生じる連結基準の不完全性については現段階では否定できるとは言い難い。この点については学習事例数を増やし、より多くの連結パターンに対応できるようにする必要がある。また、ここでは前件部パラメータとしてセグメント長、ストローク間距離、およびストローク間のなす角の正弦と余弦を与えて連結基準ファジィルールを獲得した。今回獲得したファジィ推論ルールについては、ストローク間距離 d_i となす角の余弦 $\cos\theta_i$ が連結基準に大きく影響しているようであり、ストローク間の相対位置関係が重要な要素であると考えられる。しかしながら、連結基準をファジィ推論ルールによって表現することにたいし、このパラメータ設定が適切であるか否かについては、今後、何らかのルール評価実験によって明らかにしていく必要がある。また、連筆すべきか否かのルールは、文字や部首固有のものがあると考えられ、FNNのみではルール化できない部分が存在することは否めず、この点についても今後検討する必要がある。

行列変換を用いた連筆骨格への変形処理については、取り込み数 m を閾数によって変化させることで、文字表現を多彩にできる。文献[11]において提案している手法が、1文字中の全ストロークを連結処理して草書的で文字を生成しているのにたいし、本手法はルール化された連結基準によって連結されたもののみを処理対象とし、実際の連筆文字に近い表現を実現した。ただし、閾数によって取り込み数 m を変化させることができるのはいえ、 $f(\Gamma, p)$ が最も適切な閾数であるとは言えず、また、その閾数は事前にユーザ自身が設定しておく必要がある。このことから、今後、システムが楷書文字の入力情報からこれを自動的に最適設定する機構が必要であると考えている。さらに、この変形処理では対応できていない点も専門家から指摘されていることから、別のアプローチの考案・実装を速やかに実施する必要がある。

本稿では、連筆文字の生成規則として、2ストローク間の連結基準のみを取り上げた。しかしながら、3以上の連續ストロークを必要とするルールなど、実際には他にもいくつかの書画ルールが存在する。これらをシステム上に実装し、より実際の行書体に近い表現を実現することが今後の重要な課題の1つである。また、現実社会においては、オフライン入力楷書文字、

すなわち既存の楷書体フォントなどを連筆文字フォントに変換することにも大きな需要があることから、これについても今後検討していきたいと考えている。

参考文献

- [1] 中村剛士、黒田崇、伊藤英則、世木博久. 筆記速度のファジィ評価方法を導入した毛筆文字生成システムについて. 日本ファジィ学会誌, 7(2) : 371-379, 1995.
- [2] 中村剛士、松下政親、世木博久、伊藤英則. フラクタルを用いた毛筆文字のかすれ表現について. 日本ファジィ学会誌, 8(3) : 558-566, 1996.
- [3] 中村剛士、真野淳治、世木博久、伊藤英則. 毛筆フォントの掠れ・滲み処理システムについて. 情報処理学会論文誌, 38(5) : 1008-1015, 1997.
- [4] 真野淳治、中村剛士、世木博久、伊藤英則. 毛筆書体におけるくりこみ群を用いたかすれ・にじみ表現. 情報処理学会論文誌, 38(5) : 1008-1015, 1997.
- [5] 真野淳治、江野脇宏、中村剛士、何立風、伊藤英則. 運筆状態規則を用いた平仮名毛筆文字生成について. 日本ファジィ学会誌, 11(1) : 140-148, 1998.
- [6] 増田達也、夜久正司. ユニット生成機能を持つニューラルネットワークによるファジィ推論ルールの獲得手法. 日本ファジィ学会誌, 5(2) : 348-357, 1993.
- [7] 林陽一、中井正人. ニューラルネットワークを用いたファジィプロダクションルールの自動抽出. 第5回ファジィシステムシンポジウム講演論文集, pages169-176, 1989.
- [8] L. X. Wang and J. M. Mendel. Generating fuzzy rules by learning from examples. *IEEE Trans. Syst., Man, Cybern.*, 22(6): 1414-1427, 1992.
- [9] C. T. Lin and C. S. G. Lee. Neural-network-based fuzzy logic control and decision system. *IEEE Trans. Computers.*, 40(12): 1320-1336, 1991.
- [10] 駒井鷺静. つづけ字の知識と書きかた. 東京美術, 1990.
- [11] 竹内良亘. 一筆書き曲線の変形法. 情報処理学会論文誌, 39(11) : 2997-3008, 1998.

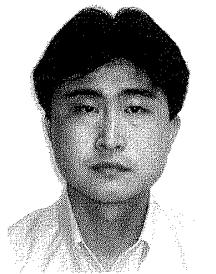
(2000年12月12日 受付)
(2001年7月3日 採録)

連絡先：

〒466-8555 名古屋市昭和区御器所町
名古屋工業大学 知能情報システム学科 伊藤研究室
中村 剛士
TEL : 052-735-5475
FAX : 052-735-5477
Email : tnaka@ics.nitech.ac.jp

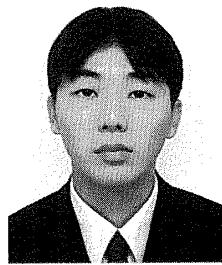
著者紹介

山田 眑嗣 [非会員]



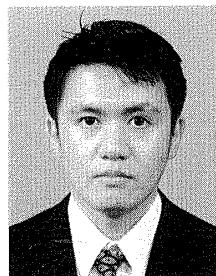
1996年名古屋工業大学知能情報システム学科卒業。1998年同大学院工学研究科博士前期課程電気情報工学専攻修了。現在同大学院博士後期課程在学中。人工知能、画像処理、動画像処理に興味を持つ。情報処理学会、IEEE Computer Society会員。

江野脇 宏 [非会員]



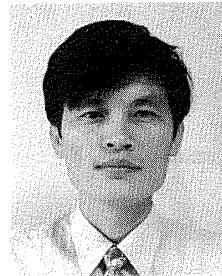
日立ソフトウェアエンジニアリング株式会社。1997年名古屋工業大学知能情報システム学科卒業。1999年同大学院博士前期課程修了。同年、日立ソフトウェアエンジニアリング株式会社入社。ファジイ理論、ニューラルネットワークに興味を持つ。在学中、毛筆文字生成システムの研究開発に従事。

中村 剛士 [正会員]



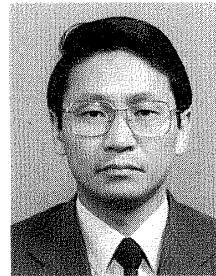
1993年名古屋工業大学工学部電気情報工学科卒業。1995年同大学院博士前期課程修了。1998年同大学院博士後期課程修了。現在同大学知能情報システム学科助手、博士(工学)。画像処理、感性情報処理、ソフトコンピューティング等に興味を持つ。電子情報通信学会、情報処理学会各会員。

何 立鳳 [非会員]



1982年中国西北輕工業学院自動制御学系卒業。同年同大学助手、1987年同講師。1997年名古屋工業大学工学研究科博士後期課程電気情報工学専攻修了。博士(工学)。1998年愛知県立大学情報科学部講師、1999年同助教授、現在に至る。人工知能、定理証明、マルチエージェント分散計算、画像処理、ファジイ推論に関する研究に従事。

伊藤 美則 [正会員]



1974年名古屋大学大学院工学研究科博士課程電気電子専攻満了。工学博士号取得。1974年日本電信電話公社横須賀研究所勤務。1985年財新世代コンピュータ技術開発機構出向。1989年名古屋工業大学教授。現在知能情報システム学科所属。この間、数理言語理論、計算機ネットワーク通信、OS、知識ベースシステムなどの研究開発に従事。電子情報通信学会、人工知能学会、形の科学学会、情報処理学会各会員。

Transforming Technique from on-line inputted *Kai-sho* style to *Ren-hitsu* style

by

Koji YAMADA, Hiroshi ENOWAKI, Tsuyoshi NAKAMURA, Lifeng HE and Hidenori ITOH

We have studied and developed some systems to generate calligraphic characters by using an electronic pen. These systems have generated *Kai-sho* style and *Hiragana* style characters, but they have been unable to generate a variety of styles of character such as *Ren-hitsu* style. In this paper, we propose one of new systems which can generate both *Kai-sho* style and *Ren-hitsu* style characters from the on-line inputted one. The system transforms *Kai-sho* style into *Ren-hitsu* style.

keywords: *Ren-hitsu* style characters, Fuzzy neural network, matrix transformation