２次元ワープを併用したオンライン英数字・数学記号認識

田畑 耕一 † 福田 亮治 ‡ 鈴木 昌和 †

† 九州大学大学院数理学研究科 〒 812–8581 福岡県福岡市東区箱崎 6−10−1
E-mail:suzuki@math.kyushu-u.ac.jp
‡ 大分大学 工学部福祉環境工学科 〒 870-1192 大分市大字旦野原 700
E-mail:rfukuda@cc.oita-u.ac.jp

あらまし 現在開発中の手書きによる数式入力インターフェースにおいて、手書き１文字認識率の向上と、書き手によらないロバストな認識を実現するために、歪んだ文字の認識が課題であった。今回、文字の歪みに強い認識手法として２次元ワープ法を用い、単純化した形でオンライン手書き文字認識に導入し、従来用いていた認識手法と組み合わせることによって、効果を上げることが出来たので報告する。また、複数の認識手法により得られたコストや順位を用いた Voting 方法についての検討も、併せて報告する。

キーワード オンライン手書き文字認識、２次元ワープ、Voting 方法

On-Line Recognition of Alphabet, Numeral and Mathematical Symbols Additionally Using Two-Dimensional Warping

Koichi Tabata† Ryouzi Fukuda‡ Masakazu Suzuki†

† Graduate School of Mathematics, Kyushu University 6–10–1 Hakozaki, Higashi-ku, Fukuoka, 812–8581 Japan
E-mail:suzuki@math.kyushu-u.ac.jp
‡ Faculty of Engineering Oita University 700 Dannoharu, Oita, 870–1192 Japan
E-mail:rfukuda@cc.oita-u.ac.jp

Abstract In our handwriting interface under development, one of our subject was the improvement of the rate of character recognition including deformed characters. In this paper, we apply simplified Two-Dimensional warping method as the robust recognition method for solving this problem, and combine it with our recognition method used so far. This paper reports this effect. Moreover, this paper also reports the examination about the voting method using the cost or order obtained from two or more recognition methods.

Keywords On-Line Hand Written Recognition, Two-Dimensional Warping, Voting Method

1　はじめに

今日のネットワークやコンピュータの加速的な進歩に伴い、その数学における用途も、単なる計算だけに留まらず様々な可能性を持っている。例えば、数式処理システムや数式のデータベースに接続した電子ボードなど、数学の授業や講義の新しいスタイルを生み出す可能性がある。また、TEX や MathML といった各種数式フォーマット、Mathematica のような数式処理システム、Word など各種ワープロに付属の数式入力機能など、コンピュータ上で数式を扱う環境も開発されて来ている。

しかし、現在、コンピュータ上で数式を扱うのはまだまだ簡単であるとは言えない。Mathematica や Word などの数式入力機能は、直感的で分かりやすいが、メニューボタンなどによる入力操作の煩わしさがスムーズな入力を妨げ、ユーザの思考の流れを中断させることが多い。TEX は数式をスムーズに入力できるが、使いこなせるようになるまで相当な修練が必要とされる。また、入力された数式の意味を一目見ただけで把握することが難しく、入力の誤りを数式を見ながら訂正することができないという事も挙げられるだろう。

この問題を解決するために、我々は、オンライン手書き文字認識を用いたユーザーにやさしい数式入力インターフェイスを開発している ([1])。このシステムはリアルタイムで手書き数式を認識し、自動的に書き換えを行うインタラクティブな手書き入力インターフェイスと、様々なフォーマットに対応し、数式の視覚的な編集を可能とするエディタからなる。

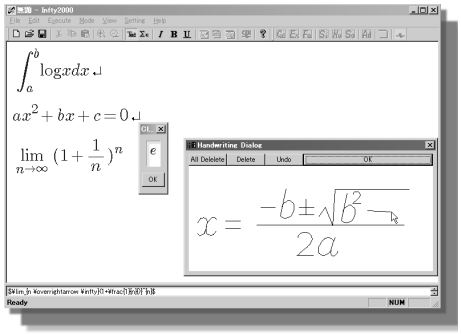

図 1: システムの外観

このシステムにおける手書き入力インターフェイスの最大の特徴は、全体の数式を書き終えてから文字認識や数式認識を行うのではなく、ペンを離す度に文字認識を実行し、文字（あるいは記号）として認識されたら直ちに、整形された文字に書き直し、位置を調整して表示してしまう点にある。これを逐次認識方式と呼ぶ ([2])。数式認識では、入力ストローク列の文字・記号の単位の切り出しや文字認識の誤り、数式構文中の位置判定の誤りが数式全体の認識を破壊してしまうことが多く、その修正が著しく困難である。また、数式の書き直しや整形に神経を裂かれることは、書き手の思考の中断を生み出すことになり、スムーズな入力の大きな障害になる。こうした困難に対応する方法として、上述の逐次認識方式を導入し、スムーズな数式入力を実現している。

したがって、この手書き入力インターフェイスにおいては、１数式要素の認識率の向上が重要な課題になってくる。これまで開発してきた手書き文字認識システムでは、方向線素特徴量による認識手法と分割ストロークによる認識手法の２つを用いて手書き文字認識を行ってきた。前者は、文字領域の５×３ブロック内における８方向の方向線素の割合を特徴ベクトルとして認識に用いる手法であり、後者は、一定の規則により分割されたストロークにおける、始点終点方向・外接矩形座標・縦横比を特徴ベクトルとして認識に用いる認識手法である。この従来手法により丁寧に書かれた文字に対しては良い認識結果を返す。しかし、書き手の書き癖などによる歪んだ文字にはあまり対応できていなかった。そこで今回、新たな方法として、２次元ワープ法という歪みに強い認識手法を組み合わせることで、解決を試みた。この手法は、その計算量の大きさによりそのままオンラインに導入することは出来ないので、計算量削減のためにアルゴリズムを簡略化して導入した。それでも十分な効果があったので報告する。また、複数認識手法による認識結果候補の Voting 方法についての検討も、併せて報告する。

2　従来の認識手法

従来は２つの認識手法を個別に用いて認識を行い、それぞれの認識結果候補における共通結果を、最終的な認識結果の上位にする方法で認識結果を投票していた。この節ではそれら２つの手法について説明する。その投票方法については第 4 節にて述べるためここでは省略する。

2.1　方向線素特徴量を用いた認識手法

この認識手法は、方向線素特徴量をベースにした手法である。文字画像は点列で与えられているので、連続した線分列とも考えられる。方向線素特徴量とは、文字画像領域のどの領域にどの方向の線分列がどれぐらいの長さ含まれているのかを数値化したものである。文字画像の外接矩形をあらかじめ設定しておいた間隔で小領域に分割し、それら小領域毎に方向線素特徴量を求めていく。

この特徴量の他、始点終点での方向、縦横比等も詳細認識のために特徴量として併せて用いている。

2.1.1　認識に用いるいくつかの特徴量

- 方向線素特徴量

  ここでは、入力された文字画像の外接矩形を、図 2 で示したように縦５×横３＝１５の小領域に分けることにする。ここで言う方向とは、図 3 のような８方向で半時計回りで定義する。したがって、各小領域の方向線素特徴量は８次元ベクトルになり、全体の画像に対しては８×１５＝１２０次元ベクトルになる。

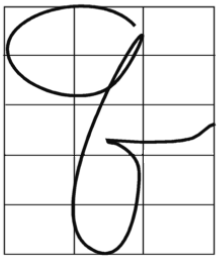

図 2: 小領域分割の例

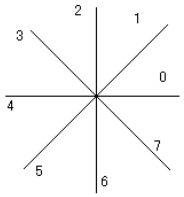

図 3: 方向

以下、詳しく述べる。

l_{i_j} を点 $P_{i_j}(x_{i_j}, y_{i_j})$ から点 $P_{i_{j+1}}(x_{i_{j+1}}, y_{i_{j+1}})$ を結ぶ線分とする。l_{i_j} の方向 (P_{i_j} から $P_{i_{j+1}}$ への向き) が 8 方向 (図 3) のどの隣り合う 2 方向 $D_i, D_{i+1} \in \{0, 1, ..., 7\}$ に挟まれるかを調べる。そして、l_{i_j} と D_i, D_{i+1} との角度差をそれぞれ θ_1, θ_2 定め、$L = |l_{i_j}|$ (l_{i_j} の長さ) としたとき、この角度差の比を用いて

$$L_i = \frac{L\theta_2}{\theta_1 + \theta_2},$$

$$L_{i+1} = \frac{L\theta_1}{\theta_1 + \theta_2}$$

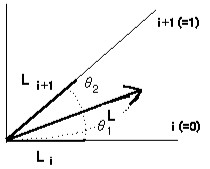

図 4: 方向の振分

のようにして l_{i_j} を 2 方向に数値化して振り分ける。l_{i_j} の中点が小領域 Q 内の点であった場合、この L_i, L_{i+1} を小領域 Q の方向線素特徴量ベクトルにおける、方向 D_i, D_{i+1} に対応する成分の値にそれぞれ加算する。この操作を全ての線分について行い、得られたベクトルの全成分の和が一定値 (ここでは２０００) になるように正規化したものを、文字画像 A の方向線素特徴量（１２０次元）とする。

- ストロークの始点、終点情報
  ストロークごとの始点・終点の座標とその方向も、ベクトルとして持つ。このベクトルの次元は文字種のストローク数に依存し、ストローク数が n ならば $n \times 6$ 次元である。

- その他の特徴量
  その他、縦横比、ストローク数などの要素も詳細認識のために特徴量に入れた。

2.1.2 辞書の作成

一文字に対して、方向線素特徴量とストローク始点・終点情報を併せたベクトルを特徴量ベクトルと呼ぶことにする。一文字は $120+6n$ 次元の特徴量ベクトルを持つ (n はストローク数)。ストローク数により特徴量ベクトルの次元数は異なる。

各文字種ごとに１００文字程度の学習データを用意する。同じ文字でも異なる書き方があれば、違う文字種データとして用意する。このデータからそれぞれの特徴量ベクトルの平均を計算したものを用意する。これを文字種のストローク数ごとに分けて並べたものを辞書とする。

2.1.3 認識結果候補の決定

以上の特徴量と辞書を用いて認識を行う。まず、入力された文字データのストローク数により、検索対象とする辞書の文字種を限定し、その中で各文字種の特徴量ベクトルと入力文字データの特徴量ベクトルとの距離差を求める。差の小さいものから 3 位までを認識結果の候補として出力する。

2.2 分割ストロークを用いた認識手法

この認識手法は、入力されたストロークを一定の規則を持ったピークポイントで分割し、その分割されたストローク毎の特徴量を求め、その特徴量の列を用いて認識を行う手法である。

2.2.1 ストロークの分割規則と基本ストロークの定義

入力されたストロークは、縦のピークポイントで分割し、さらに横のピークポイントで入射角反射角の差が 90 度より大きい場所では分割する。

縦のピークポイントとは、点列の y 座標値の変化を時間順に見たとき、その正負が入れ替わる点のことである。正負が入れ替わる部分で y 座標値が同じ点が続く場合は、それらの点を結んだ x 軸に水平な線分の中点をピークポイントとする。まずこの点でストロークを分割する。

同様に考えて横のピークポイントは x 座標値の変化の正負が入れ替わる点である。そして、この横のピークポイント $Peak(x) = P_{i_j}$ であったとき、ベクトル $\overrightarrow{P_{i_{j-1}}P_{i_j}}$ とベクトル $\overrightarrow{P_{i_j}P_{i_{j+1}}}$ のなす角 θ_x が $\theta_x > \frac{\pi}{2}$ の場合、$Peak(x)$ によりストロークを分割する（図 5 参照)。

この規則により分割されたストロークの１単位を基本ストロークと呼ぶことにする。

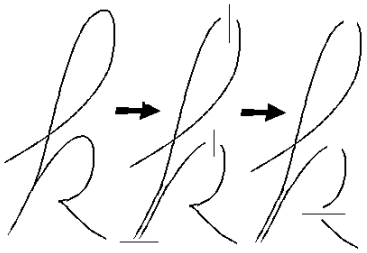

図 5: ストローク分割方法

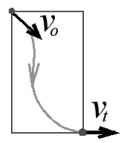

図 6: 始点終点方向

2.2.2 認識に用いるいくつかの特徴量

基本ストロークに対して、始点、終点での方向 (図 6)、外接矩形、縦横比を求めて、ベクトルを生成する。

このベクトルの次元は基本ストローク数を n とすると、$n \times 7$ 次元である。このベクトルを全ストロークに対してストローク順に並べたものを、この認識手法の特徴ベクトルとする。１文字データのストローク数を N とし、ストローク s_i の基本ストローク数 (ストローク分割数) を d_i とすると、特徴ベクトルの次元は $\sum_{i=1}^{N} d_i \times 7$ で表

される。

また、基本ストロークは、元のストロークの縦のピークポイントでは必ず分割されているので、y 方向に関しては単調増加 (アップストローク) か単調減少 (ダウンストローク) のどちらかになっている。この性質を用いて基本ストロークを２値 $(0,1)$ で表し、ストロークを記号列化する。これをストローク順に並べたものを、１文字データの大分類記号列と呼ぶことにする。この大分類記号列を用いることにより、辞書検索の効率化を行う。

2.2.3　辞書の作成

各文字種ごとに１００文字程度の学習データを用意する。このとき、同じ文字種内で、予想し得る大分類記号列がほとんど出つくすよう、注意して学習データを集める。こうして集めた学習データに対して、同じ大分類記号列かつ同じ文字種であるものについて、特徴量の平均を取る。これを全ストローク分割数ごとに並べたものをこの認識手法の辞書とする。

2.2.4　認識結果候補の決定

以上の特徴量と辞書を用いて認識を行う。まず、入力された文字データの大分類記号列により、検索対象とする辞書の要素を限定し、その中で各文字種の特徴量ベクトルと入力文字データの特徴量ベクトルとの距離差を求める。差の小さいものから３位までを認識結果の候補として出力する。もし大分類記号列が見つからなかった場合、同じストローク分割数である全てのデータに対して、特徴量ベクトルの距離差を求める。この距離差にさらにいくらかの値 (ペナルティ) を追加し、上と同様に距離差の小さいものから３位までを認識結果の候補として出力する。

3　２次元ワープ法を用いた認識手法

この手法は内田、迫江の論文（[3],[4]）を参考にした手法である。２次元ワープとは、２画像の最大一致を与える２次元－２次元写像のことを呼ぶ。パターン認識の立場から見れば、２次元ワープは画像の変形に適応可能なマッチング処理であると言える。この２次元ワープの最適化問題、すなわち２画像の最大一致を与える２次元ワープを探索する問題は、動的計画法（ＤＰ）に基づくアルゴリズムによって解くことができる。しかし、論文 [3] におけるＤＰの計算量は、最悪の場合指数オーダーとなり、オンラインのシステムに組み込むには実用的でなく、このままでは用いることはできない。

そこでこの手法の文字の歪みに強いという特色を出来るだけ残しつつ、単純化した形でオンラインに導入する。以下ではその手法を説明する。

3.1　認識に用いる特徴量の設定

まず、文字データを $M \times M$ の小領域に分ける。本論文では $M = 10$ としている (図 7 参照)。この小領域ごとの方向線素特徴量（2.1.1 を参照）を８方向 (図 3) にて求める。

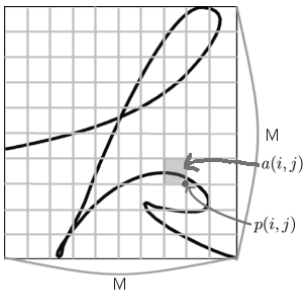

図 7: 小領域に分割された例

このとき、格子点を $p(i,j),(i,j = 0,1,...,M)$ とし、$p(i-1,j-1),p(i-1,j),p(i,j-1),p(i,j)(i,j \geq 1)$ の４点で囲まれた領域の方向線素特徴量を $a(i,j)$ で表すことにする。$a(i,j)$ は８次元ベクトルである。

これ以降、文字データ A をただの点列ではなく画像として捉えると、理解しやすい。この場合、$a(i,j)$ を画素と思うことにする。この特徴量を元に、２画像

$$A = \{a(i,j)|i,j = 1,2,...,M\},$$

$$B = \{b(x,y)|x,y = 1,2,...,M\}$$

との２次ワープの最適化問題を考える。

3.2　ピボットの紹介と設定方法

画像を横に N 分割する線分を $l(i),(i = 0,1,...,N)$、その分割された段を $L(i),(i = 1,2,...,N)$ とする (図 9)。$L(i),(i \geq 1)$ は $l(i-1)$ と $l(i)$ に挟まれた画像である。ここでは画像の縦の最小単位が M であるため、N には $N|M$ の条件が付く。本論文では $N = 5$ としている。

この $l(i)$ を折り曲げて $L(i)$ を画像 B 上に重ね合わせることを考える。この折り曲げを行う $l(i)$ 上の点を中間ピボットと呼び、(i,j_2) と表す。また、$l(i)$ の左端 (i,j_1) を左ピボット、右端 (i,j_3) を右ピボットと呼ぶ。ここで $0 = j_1 < j_2 < j_3 = M$ とする。

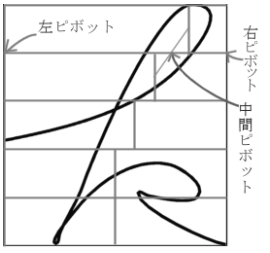

図 8: 入力パターン (A)

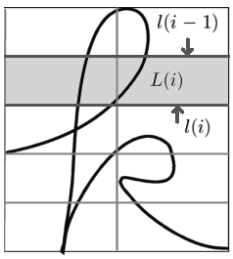

図 9: 標準パターン (B)

ここでは入力パターン A が、対象パターン B であると見たときの A の中心線を予測して、その中心線と $l(i)$ との交点を中間ピボットとして定める事とする (図 8)。

中心線の予測は、入力パターン A の段 $L(i)$ ごとに行う。まず $L(i)$ に含まれる方向線素特徴量を用いて中心線

$lp(i)$ を予測する。簡単のために、画素の１単位を

$$a_l(i,j) = \sum_{k=(\frac{M}{N}-1)i}^{\frac{M}{N}i} a(k,j)$$

として考える。$\frac{M}{N}$ は $L(i)$ に含まれる画素の行数であるから、$a_l(i,j)$ は $L(i)$ 画像を横方向に M で分割したときの、j 列目の画素の和 (方向線素特徴量の和) である。

$L(i)$ を左右に分割する線分 $lp(i,j)$ を、格子点 $p((\frac{M}{N}-1)i,j), p(\frac{M}{N}i,j)$ を結ぶ線分とし、j を $1 \leq j \leq M-1$ の範囲で動かしたとき、$lp(i,j)$ で区切られた左右の領域の方向線素特徴量 $L_{left}(i,j), L_{right}(i,j)$ は、

$$L_{left}(i,j) = \sum_{k=1}^{j} a_l(i,k)$$
$$L_{right}(i,j) = \sum_{k=j+1}^{M} a_l(i,k)$$

$(1 \leq j \leq M-1)$ と、対象パターン B の、i 段目の左右の小領域の方向線素特徴量 $B_{left}(i), B_{right}(i)$ との差の和、

$$B_{left}(i) - L_{left}(i,j) + B_{right}(i) - L_{right}(i,j)$$

が最も小さくなるような $j = \acute{j}$ を各段で求め、

$$lp(i) = lp(i,\acute{j})$$

とする。

以上で求まった $lp(i), lp(i+1), (1 \leq i \leq N-1)$ の中点間を結んだ線分と $l(i)$ との交点を、$l(i)$ 上の中間ピボットとして定めるようにする。$i=0$ の場合は、$l(1)$ 上の中間ピボット $(1, j_2)$ と $lp(1)$ の中点を結んだ直線と $l(0)$ との交点、$i=N$ の場合は、$l(N-1)$ 上の中間ピボット $(N-1, j_2)$ と $lp(N)$ の中点を結んだ直線と $l(N)$ との交点、をそれぞれ中間ピボットとして定める。

3.3 ワープ関数

これら、左、中間、右ピボットのワープによる像を用いて、各段 $L(i)$ のワープを考える。

その前に準備として、次の定義をする。

格子上の異なる４点が与えられたとき、この４点を結んでできる四角形領域 S における方向線素特徴量は

$$\sum_{p_c(i,j) \text{ in } S} a(i,j)$$

($p_c(i,j)$ は画素 $a(i,j)$ の領域の中心点であり、p in S とは S 内に点 p が含まれることを意味する) で与えられるものとする。

いま、$L(i)$ を $L_{left}(i)$ と $L_{right}(i)$ に分けて考える。$L_{left}(i)$ は、$l(i-1)$ 及び $l(i)$ 上の左、中間ピボットの４点で表される領域であるが、この領域における方向線素特徴量により $L_{left}(i)$ を表現することにする。つまり、$L_{left}(i)$ は８次元ベクトルと見なす。$L_{right}(i)$ についても同様である。

ここから、$L_{left}(i)$ のワープによる像 $\acute{L}_{left}(i)$ を、各ピボットの像によって囲まれた領域の方向線素特徴量と定める。すなわち $l(i)$ 上の、左、中間、右ピボットのワープによる像をそれぞれ $(x_1^i, y_1^i), (x_2^i, y_2^i), (x_3^i, y_3^i)$ と表し、４点 $(x_1^{i-1}, y_1^{i-1}), (x_2^{i-1}, y_2^{i-1}), (x_1^i, y_1^i), (x_2^i, y_2^i)$ による四角形領域を $S(i, left)$、４点 $(x_2^{i-1}, y_2^{i-1}), (x_3^{i-1}, y_3^{i-1}), (x_2^i, y_2^i), (x_3^i, y_3^i)$ による四角形領域を $S(i, right)$ とすると、

$$\acute{L}_{left}(i) = \sum_{p_c(i,j) \text{ in } S(i,left)} a(i,j)$$
$$\acute{L}_{right}(i) = \sum_{p_c(i,j) \text{ in } S(i,right)} a(i,j)$$

と表せることになる。

この $\acute{L}_{left}(i), \acute{L}_{right}(i)$ により $L(i)$ のワープによる像 $\acute{L(i)}$ を表す。

3.4 目的関数と画像間の距離

画像 A の段 $L_{left}(i), L_{right}(i)$ のワープによる像と、画像 B の対応する段 $B_{left}(i), B_{right}(i)$ との距離を、前節の表記を用いて次式で定義する。

$$\hat{d}(\acute{L}_{left}(i), B_{left}(i)) = \|\acute{L}_{left}(i) - B_{left}(i)\|$$
$$\hat{d}(\acute{L}_{right}(i), B_{right}(i)) = \|\acute{L}_{right}(i) - B_{right}(i)\|$$

これより、$L(i)$ のワープによる像 $\acute{L}(i)$ と $B(i)$ の距離を

$$d(i) = \hat{d}(L_{left}(i), B_{left}(i)) + \hat{d}(L_{right}(i), B_{right}(i)) \quad (1)$$

と定める。

全体の画像に対してのワープの実行においては、距離 d を全ての段 i について加算した

$$\sum_{i=1}^{M} d(i) \quad (2)$$

を目的関数とし、これを最小化するワープ関数を求める。目的関数 (2) の最小値を $D(A,B)$ と表すことにする。$D(A,B)$ は２次元ワープにより２画像 A, B をマッチングしたときの画像間の距離に相当する。

3.5 ピボットの制約条件

ワープに適切な自由度を与えるために、ピボットの像に関して以下のような制約条件を考える。

左右のピボットは縦方向のみにワープされる。すなわち、

$$x(i,j_1) = 1\ ,\ x(i,j_3) = M \quad (3)$$

とする。

また、縦方向の像については、

$$i - w \leq y(i, j_1) \leq i + w$$
$$i - w \leq y(i, j_3) \leq i + w \quad (4)$$

のような制限を設ける。本論文では計算量の軽減のため $w = 1$ としている。

中間ピボットについては縦横両方向のワープを考える。その像は、

$$i - w \leq x(i, j_2) \leq i + w$$
$$i - w \leq y(i, j_2) \leq i + w \quad (5)$$

のような制限を設ける。

3.6 動的計画法を用いた最適化アルゴリズム

２次元ワープの最適化問題は、制約条件 (4)(5) の下での目的関数 (2) の最小化問題であり、いわゆる多段決定過程を用いて記述できる。このときの段は画像 A の $L(i)$ に対応する。

$L(i)$ のワープによる像は、$L(i)$ を挟む $l(i), l(i-1)$ 上の左右、中間ピボットのワープによる像で決定される。そこで、$l(i)$ 上のピボットの像の組を $p_{l(i)}$ と表すと、多段決定過程の第 i 段に属する状態は ($p_{l(i-1)}$, $p_{l(i)}$ $|i$) の組により表される。$p_{l(i)}$ は (4)(5) を満たす範囲で存在するので、この状態は (4)(5) を満たす範囲で存在する。$p_{l(i)}$ の (4)(5) を満たす範囲での集合を $P_{l(i)}$ としておく。

段の推移に伴う状態遷移に応じて局所コスト (1) を加算していく。第１段から第 N 段までの状態遷移系列のうち、局所コストの累積値が最小となる系列が、２次元ワープ問題の解に対応する。この最適化状態遷移系列は図 10 の動的計画法 (ＤＰ) アルゴリズムで探索できる。ここで、$g(p_{l(i-1)}, p_{l(i)}|i)$ は、状態 ($p_{l(i-1)}$, $p_{l(i)}$ $|i$) に至るまでの、過去のワープ経歴に関する局所コストの最小累積値とする。

これにより画像 A, B 間の距離 $D(A, B)$ を求める。

3.7 計算量とその効率化 (ビームサーチ)

本節では、図 10 のＤＰアルゴリズムの計算量が画像サイズ M に関して多項式オーダーになることを示す。この計算量は、ＤＰ漸化式の計算回数と、ＤＰ漸化式一回当たりの計算量の積で与えられる。

ＤＰ漸化式の計算回数は、図 10 の step5 の繰り返し回数に等しく、全状態数に等しい。$W = 2w + 1$ と表すと、両端ピボットには W、中間ピボットには W^2 の自由度があるから、各 i 段に属する状態数は $W \times W^2 \times W = W^4$ となる。これと段数 N の積により、漸化式計算回数 $O(NW^4)$ が得られる。

ＤＰ漸化式１回あたりの計算量は局所コスト (1) の計算量と最小値選択回数の積となる。前者は $O(\frac{M^2}{N})$ であり、後者は１状態から遷移可能な状態数に等しいので、$W \times W^2 \times W = W^4$ となる。よって漸化式１回当たりの計算量は $O(W^4 \times \frac{M^2}{N})$ となる。

(初期値設定)

1: **for all** $p_{l(1)} \in P_{l(1)}$

2: $g(p_{l(0)}, p_{l(1)}|1) := d(1)$

(ＤＰ漸化式)

3: **for** $i := 2$ **to** N **do**

4: **for all** $p_{l(i)} \in P_{l(i)}$

5: $g(p_{l(i-1)}, p_{l(i)}|i) = \min_{p_{l(i-1)} \in P_{l(i-1)}} \{d(i) + g(p_{l(i-2)}, p_{l(i-1)}|i-1)\}$

(終了)

6: $D(A, B) := \min_{p_{l(N)} \in P_{l(N)}} g(p_{l(N-1)}, p_{l(N)}|N)$

図 10: ＤＰアルゴリズム

以上よりＤＰアルゴリズム全体の計算量は $O(M^2W^8)$ であり、M 及び W に関して多項式オーダーとなる。

ここからさらに計算量を軽減するために、ビームサーチという手法を用いる。

ビームサーチとは、動的計画法において、最適経路としての可能性が低いと判断されたものを以後の処理から除外することで探索空間を圧縮し、計算量とメモリ量の低減を同時に実現できる効率化法である (論文 [4])。ここでは、ある段において、一つ前の段から遷移できる状態を、累積コストの少ない順に１０位までの状態に限定した。これにより計算量は $O((10 \times \frac{M^2}{N})NW^4) = O(M^2W^4)$ まで軽減される。

3.8 辞書の作成

各文字種ごとに１００文字程度の学習データを用意する。同じ文字でも異なる書き方があれば、違う文字種データとして用意する。各文字種ごとに、点列の座標値の平均をとり、平均的なきれいな文字を作成する。この平均的な文字に対して方向線素特徴量 ($5 \times 2 \times 8 = 80$ 次元) を抽出する。これを全文字種に対して用意したものを、ストローク数により分けて並べ、辞書を作成する。

3.9 認識結果候補の決定

まず、入力されたストローク数 n の文字データ A から抽出された $10 \times 10 \times 8 = 800$ 次元 のベクトルを用いて、外接矩形を 5×2 に分割した場合の方向線素特徴量の $5 \times 2 \times 8 = 80$ 次元 ベクトルを求め、辞書中の n ストローク文字種の特徴ベクトルの中で、ベクトル差の小さいものから最高１０位までを取り出す。この１０位までの候補 $B(i)$ に対して、ＤＰマッチングを行い、画像間の距離 $D(A, B(i))$ が小さいものから順に３位までを認識結果候補として出力する。

4 認識結果候補の Voting 方法

この節では、従来の認識結果の複合方法を述べた後、複数の認識手法による認識結果の Voting 方法をいくつか提案する。そして第 5 節にて、単独の認識手法の場合と複数の認識手法を用いた複合認識の場合の文字認識実験をそれぞれ行い、認識手法が 1 つの場合と複数の場合、それぞれの Voting 方法の性能等を比較・検討する。

以下では簡単のために、方向線素特徴量による認識手法を L-Method、分割ストロークによる認識手法を D-Method、2 次元ワープによる認識手法を W-Method と呼ぶことにする。

4.1 従来の認識結果複合方法 (L-Method と D-Method のみ用いる)

L-Method 、D-Method による第 i 位の認識コストをそれぞれ $C_l(i), C_d(i)$ とする。このとき、それぞれの認識結果の複合コスト $M_l(i), M_d(i)$ を

$$M_l(i) = \frac{C_l(i)}{C_l(3)} \ , \ M_d(i) = \frac{C_d(i)}{C_d(3)}$$

と定めておく。

L-Method の候補に対し、D-Method に共通の候補数が

1. 1つ → その候補を複合認識結果の 1 位として出力
2. 2つ以上 → 共通の候補同士の複合コストの和を計算しその和が小さいものから順位をつけ出力。
3. 共通の候補が無い場合は L-Method の手法による結果をそのまま複合認識結果とする ([2])。

4.2 順位に応じた Voting 方法

この Voting 方法は、各認識候補に順位に応じた得点を持たせ、同じ認識候補同士で得点を足し合わせて (投票して) 高い得点の順に複合認識結果を決める方法である。ここでは、認識候補が N 位まで用意されている場合に、認識手法によらず第 i 位の候補に $(N - i + 1)$ 点の得点を持たせることにする ([5])。後述の実験では $N = 3$ としている。

4.3 信頼度を用いた Voting 方法

ここで言う信頼度とは、認識結果のコストが分かっているときの、その認識結果が正解である条件付確率である。

ある文字データ d の文字 c としての文字認識コストを $q(d, c)$ とする。コストは広い範囲に離散的に値を持つので、一定間隔の区間内のコストを同一視することにする。具体的には、

$$\acute{q} = \left\lfloor \frac{q}{M} \right\rfloor$$

($\lfloor x \rfloor$ は $y \leq x$ である最大の $y \in \{\mathbb{N} \cup 0\}$) として、これ以降ではコスト q は $\acute{q}$ に変換して考える。M は、それぞれの認識手法毎における正解を与えるコストの最大値を $Max(cost)$ とすると、

$$M = \left\lfloor \frac{Max(cost)}{50} \right\rfloor$$

アルファベット	a,b,$\cdots$,z,A,B,$\cdots$,Z
数字	0,1,2,$\cdots$,9
ギリシャ文字	$\alpha, \beta, \gamma, \delta, \varepsilon, \theta, \lambda, \mu, \nu, \pi, \phi, \varphi, \omega, \Phi, \Psi, \Omega$
特殊記号	$\sum, \infty, \int, \sqrt{}, comma, period$
2 項演算子	$+, -, \times, \div, \pm, \mp, \mid, /, \backslash, \oplus, \cap, \cup$
関係演算子	$=, \neq, \equiv, \not\equiv, <, >, \leq, \geq, \subset, \supset, \in, \ni, \notin, \rightarrow$
連文字	lim,log,sin,cos,tan
括弧類	$(,), \{,\}, [,]$

表 1: 認識対象文字

で表される定数である。

ここで $d[i]$ を独立試行で得られたデータ列で、考察する事象に対し出現頻度で確率を近似するに十分な数があるものとする。この $d[i]$ を用いて、

$$P(result(d) = c | \acute{q} = \acute{Q}) = \frac{\#\{i : result(d[i]) = c \text{ かつ} \acute{q}(d[i], c) = \acute{Q}\}}{\#\{i : \acute{q}(d[i], c) = \acute{Q}\}}$$

で近似したものを信頼度とし、これを各認識手法、各文字種 $c[j]$ ごとに計算してテーブルとして用意しておく。

各認識手法・各認識結果候補は、そのコストと認識結果によりあらかじめ用意されたテーブルから信頼度を得る。この信頼度を持ち票とし、投票をして得点の高い順に 3 位までを候補として出力する。

4.4 信頼度を用いた Voting 方法

4.3 の Voting 方法に対して、さらに信頼度に順位に応じた重みをつけた持ち票による投票も考える。具体的には、認識候補が N 位まで用意されている場合に、認識手法によらず第 i 位の信頼度に $(N - i + 1)$ をかけたものを得点として持たせて投票する。後述の実験では $N = 3$ としている。

5 認識実験

5.1 認識対象文字と注意する書き方

今現在認識対象としている文字を表 1 で示す。基本的に高校、大学初年度程度の数式に現われる文字・記号を対象としている。特に書く頻度の少ないギリシャ文字などは、対象から外してある。

違う文字同士で区別がつきにくいいくつかの文字に関しては、書き方に制限を加える (表 2)。複数のストロークによって構成される文字については、基本的に書き順は自由である。O や ∞ を書く場合の回転方向は問わない。

5.2 実験データの解説と実験方法

以上の１２１文字について、ペンディスプレイを用いて１０人に 3 回ずつ書いてもらったデータを実験データとした。

このデータについて

$$P_1 = \frac{\text{第 1 位が正解である数}}{\text{全データ数}} \times 100$$

対象文字	制限
C, S (大文字)	はじめに縦線をつける
U, V, W (大文字)	上に横線をつける
X (大文字)	上下に横線をつける
P (大文字)	下に横線をつける
Z (大文字)	両端と真ん中に線をつける
q (小文字)	最後にはねをつける

表 2: 注意する書き方

$$P_3 = \frac{\text{第３位までに正解がある数}}{\text{全データ数}} \times 100$$

を、各認識手法ごと、従来手法 (4.1 の手法)、及びその他の Voting 方法 (4.2-4.4) について求める。4.2-4.4 については２次元ワープを含めない場合と含める場合の両方を求める。

5.3 実験結果と考察

- 認識手法を単独で用いた場合の認識率

認識手法	$P_1(\%)$	$P_3(\%)$
L-Method	92.5	97.2
D-Method	87.6	95.5
W-Method	80.3	92.5

- 従来手法及び各 Voting 方法を用いた場合の認識率

		$P_1(\%)$	$P_3(\%)$
4.1	従来手法	93.8	98.1
4.2	(L+D)	93.8	97.9
	(L+D+W)	95.0	98.1
4.3	(L+D)	95.7	99.0
	(L+D+W)	96.6	99.3
4.4	(L+D)	96.8	99.0
	(L+D+W)	97.7	99.4

２次元ワープを加える前と後での認識率の比較により、２次元ワープを組み込むことが有用であることが分かる。

ここで、最も良い結果だった 4.4 の手法による、W-Method により効果があった具体的な例を挙げてみる。

図 18 は、非常に歪んだ文字の例である。このような文字は、２次元的な伸縮を持つため、ストロークを時系列として眺めただけではその特徴をうまく抽出することは難しい。W-Method は２次元的な情報を用いるため、このような文字でも比較的上位に正しい結果が出力される可能性がある。実際このデータは W-Method の１位に正しい結果が出力されたため、その他の２つの認識手法による結果が悪かったが、複合結果では１位になっている。

図 19 は、左利きの書き手による左に傾いた文字の例である。従来の認識手法においては、左利きの書き手による文字はほとんど想定されていなかったため、左傾きの文字の認識はあまり良いとはいえなかった。しかし、W-Method の導入により、左傾きの文字は左に歪んだ文字として捉えられるので、対応が出来る。このデータの複合結果１位も「m」であった。

図 11: W-Method 効果１

図 12: W-Method 効果２

6 結論

歪みに強い認識手法の導入という観点から、２次元ワープ法による認識手法を導入し、その効果を見るために様々な Voting 方法を用いて認識実験を試みたが、どの Voting 方法を用いても、従来の認識手法に比べ良い結果が得られている点から、２次元ワープ法による認識手法がオンライン手書き文字認識にも有用であることが言える。

また、これらの Voting 方法は、本論文に限らず一般的な Voting 方法として用いることができる可能性を持ち、多方面での利用が期待される。

参考文献

[1] H.Okamura, T.Kanahori, W.Cong, R.Fukuda, F.Tamari and M.Suzuki,"Handwriting Interface for Computer Algebra Systems", *Proc. 4th. Asian Technology Conference in Mathematics*, pp.291-300. December. 1999.

[2] T.Kanahori, K.Tabata, W.Cong, F.Tamari, and M.Suzuki, "On-Line Recognition of Mathematical Expressions Using Automatic Rewriting Method", *Advances in Multimodal Interfaces —ICMI2000, Lecture Notes in Computer Science 1948,Springer,* pp. 394-401, 2000.

[3] 内田誠一, 迫江博昭, "区分線形２次元ワープ法の検討", 電子情報通信学会論文誌 (D-II), vol.J83-D-II, no.12, pp.2622-2629, Dec. 2000.

[4] 内田誠一, 迫江博昭, "動的計画法に基づく単調２次元ワープ法の検討", 電子情報通信学会論文誌 (D-II), Vol.J81-D-II, No.6 pp.1251-1258, June. 1998.

[5] 中野康明, "文字認識・文書理解の最新動向 [V] —認識後処理と多数決処理— ", 電子情報通信学会誌, Vol.83, No.6 pp.467-471, June. 2000.