

手書き入力による数式認識システム

村瀬敦史, 福島英洋, 曾谷俊男, 中川正樹
(東京農工大学 工学研究科 情報工学講座)

本稿では、オンライン手書き入力方式による数式認識システムにおけるHI部および文字認識部について述べる。数式などのように、文字の大きさや位置に構造上の重要な情報が表現されている対象に対しては、手書き入力の利点は大きい。本稿ではまず、筆記時の思考の中断と修正の時間に着目し、ユーザの利用形態に応じた創造系と複写系という二つのタイプのHIの設計を述べる。創造系とは、数式を解いたり、展開したりする場合に対応し、複写系とは、参考書などから写したり、作成した数式を消書する場合を指向している。次に、我々の有するオンライン手書き文字認識システム JOLIS-2E の数式認識のための改良について述べる。

A System for Recognizing Handwritten Mathematical Expressions

Atsushi Murase, Hidechiro Fukushima, Toshio Souya and Masaki Nakagawa

Department of Computer Science,
Tokyo University of Agriculture and Technology,
Koganei-shi, Tokyo, 184 Japan

This paper describes a design of human interface and improvement of on-line recognition to input mathematical expressions through handwriting. Handwriting interface has a great advantage over other interfaces when important information on the structure of and among objects to be put into is expressed in their size and position. Such information must be otherwise specified by commands or extra codes.

Two types (creative-type and copy-type) of handwriting interface are designed, taking both interruption of thinking and labor for corrections into account. When a user is working on a tablet to solve or transform mathematical expressions, his thinking must not be interrupted and the interface should be natural and creative. On the other hand, the interface should be efficient for copying such expressions from a book or whatever else.

The paper also presents modifications applied to the on-line character recognition system, JOLIS-2E, to adjust it for mathematical expressions.

1. はじめに

現在非常に多くの人々が文書作成をワードプロセッサ（以下ワープロ）を用いて行うようになってきている。確かにワープロは文書の訂正、管理、再利用などの観点からいえば優れたものであるといえるだろう。しかし、その反面、図表や数式など作成しづらいもの、あるいは作成するのに非常に時間のかかるものが存在しているなど、いくつかの問題点も明らかになってきた。これらの問題への対応はもちろん試みられているが、使い勝手の問題は依然として残されている。一方、ワークステーションやDTPをうたうワープロでは、数式や図表の整形ソフトなどが提供されている。これらを利用すると、写植に近い出力が得られる反面、整形コマンドの人力がわずらわしく、また、出力イメージとの対応がとりにくい問題がある。少なくとも、数式を解いたり、展開するために利用すべきものではない。

最近では、数式処理ソフトウェアがパーソナルコンピュータ上でも利用できるまでになっている。入力インタフェースの問題が解決されれば、数式の処理、整形出力が楽になるばかりか、数式のCAIなどにも利用できるであろう。

我々は、性能の向上が進んでいる表示一体型タブレットの利用を前提に、手書き入力インタフェースの実現を進めている。本稿では、その中で特に数式入力を取りあげる。次章で現在ある文書作成システムの問題点を整理することにより、手書き数式入力インタフェースの目指すべきものを考察する。3章では、オンラインパターン認識技術の現状を踏まえ、手書き数式入力インタフェースの設計について述べる。4章では、数式に用いられる文字パターンの性質について考察し、我々が有するオンライン文字認識システムJOLIS-2Eを数式専用に改良する実験について述べる。

2. 数式作成における手書き入力方式の適合性

入力方式を手書きにすることにより、ワープロなどの文書作成システムにおける、数式作成時に生じる問題点を本当に解消できるのだろうか。ここでは、

- ・文書作成システムの問題点とは具体的に何なのか
- ・手書き入力方式はこの問題点を解消できるのか

という二点について考察してみる。

2.1 文書作成システムの問題点

ひとくちに文書作成システムといっても、それらは目的に応じ、いくつかは大別される。これには大きく分けて三つが存在していると考えられる。次にそれらを挙げる。

[タイプA：数式入力専用のソフト・機能がついていない従来のワープロ]

- ・文字の大きさ、配置など出力のレイアウトに関する機能はそれほど充実していない
- ・文章と数式、図表を同じフィールドで扱う
- ・WYSIWYGがある程度可能
- ・数式や図表よりも文章をその対象としている

[タイプB：数式入力専用のソフト・機能がついているワープロ、DTP]

- ・上記[A]のタイプに数式や図表専用のソフト・機能を付加してある
- ・文章と数式、図表を別のフィールドで扱う
- ・文書の編集機能に優れている
- ・個々の数式や図表に関しては品質の高い出力を得ることができる
- ・WYSIWYG方式

[タイプC: \mathbb{N} などの出力システム]

- ・整形にはプログラムを用いる
- ・出力のレイアウトに関する機能が充実している

それぞれのシステムを用いて数式を作成する場合における共通の特徴と、入力方式と出力品質に関する個々の特徴を考えると、表1のようになる。

表を見てわかるように、タイプCのシステムを用いれば、高品質の出力を得ることができ、文書の管理も容易に行える。つまり、これらが文書作成システムの長所といえる。では、逆に短所は何であろうか。次にそれらを列挙する。

- ・入力時に思考の中断が生じる
- ・文字の配置，大きさに制限がある
- ・入力する際に，必要な知識を学習する必要がある
- ・整形コマンドの挿入やモード変換などの手間を要する
- ・ユーザが数式の構造を把握しづらい

使用するシステムのタイプにより，上に挙げた問題点のすべてが生じるわけではないが，少なくとも一つ以上の問題点を抱えている。これらの問題点は，すべて数式を入力するときにかかるものである。

表1 数式に関する文書作成システムの特徴

		入力方式	出力品質
個々の特徴	タイプA	文字の配置の制限により，ユーザが工夫して構造を示すレイアウトを作成しなければならない。	数式本来のレイアウトにならない場合が多い
	タイプB	数式入力用の機能を使用することにより，ユーザが個々の数式のレイアウトを考えずに入力することはできるが，その機能の操作方法を学習する必要があるし，綺麗な出力を得るためには多くの手間を要する。	複数の数式を統一的に扱うことは難しいが，個々の数式に関しては，正しいレイアウトの出力を得ることができる。
	タイプC	ユーザが数式の構造を把握しづらい，数式整形言語を学習しなくてはならない，数式の種類によっては，非常に多くの整形コマンドの挿入が必要な場合がある。	統一的な高品質の出力を得ることができる。
共通の特徴		入力時に思考の中断が生じる。 文書の管理が容易に行える。	

2. 2 手書き入力方式

上で挙げたような文書作成システムにおける問題点は，手書き入力方式にすることで解消するであろうか。手書きの長所短所を考えてみると，表2のようになる。

概して，手書きは記入時において優れているといえる。これは，次に述べる理由によるものと思われる。

そもそも数式は，文字の大きさや配置により，その構造を示している。つまり，数式には構造を示すための表記上の規則が存在しているわけである。手書きの数式は，この表記規則に沿って書かれている。人間は，幼いときからこの表記規則を学び，使用してきた。つまり，人間にとってこの表記規則に沿って書く数式こそが，特別な訓練を必要としない，一番自然で容易に理解でき

表2 手書きの長所短所

長所	<ul style="list-style-type: none"> ・思考の中断が少ない ・文字の配置，大きさに制限が無い ・数式の構造を把握し易い ・特別な学習を必要としない ・記入時に必要な手続きはない
短所	<ul style="list-style-type: none"> ・できあがり汚い ・文書の管理が大変

るものなのである。更に手書きでは、記入するのに手続きがあるわけでもないので、思考の中断が少ない。我々は、これが手書きの非常に大きな長所と考えている。人間が文字を書くときは、創造しながら書く場合と、あるものを写す場合に分かれる。後者の場合は、思考の中断が生じてもそれほど障害とならないが、前者の場合は、その創造に多大な影響をおよぼす。これらが、記入時に手書きが優れている理由である。

2. 1で述べたように、文書作成システムは、文書の管理と出力という面で優れており、入力にその問題点を抱えていた。一方手書きは、入力面で優れていることを上に示した。したがって、数式認識システムの入力方式として、手書きを用いることにより、従来の文書作成システムの問題点を解消できると考えることができる。ただし、パターン認識の能力を十分考慮したシステム設計が必要になる。

3. ヒューマンインタフェースの設計

3. 1 構造化処理

“数式を認識する”とは、どういうことだろうか。数式は、文字の集合体で表現される。そして、集合の要素となっている文字は、ばらばらに存在するのではなく、すべて構造を示すための鎖のようなものでつながれている。つまり、“数式を認識する”とは、入力された文字列の大きさや位置より、その数式の構造を作成し、計算機内に文字列と共に格納する処理のことをいう。この処理を構造化処理と呼ぶ。

3. 2 構造化処理方法の分類

上で述べた構造化処理には、構造の作成方法の違いで次に挙げる二つのタイプが考えられる。

- ・構造抽出型 : 入力された文字の大きさや配置から、構造を抽出する方法
- ・構造定義型 : 構造を先に定義して、後から各部分に内容を挿入する方法

構造抽出型とは、何の制限もなくタブレット上に書かれた数式を、その文字の大きさや位置からすべてシステムがその構造を作成する方式である。例えば、“ Σ ”が書かれたときに、上と下に書かれた小さな文字列と右に書かれた文字列とを組み合わせて、ひとつのシグマの数式を構造化したり、ある文字、例えば“a”が書かれ、その後右上に小さく“b”と書かれたならば a^b とシステムが自動的に構造化するものである(図1(a)参照)。この方法は、通常我々がペンで紙面上に数式を書いている状態である。この方法であれば、ユーザがタブレットに数式を書いているときに思考の中断が入ることが少ない。しかし、認識系の負担が大きくなり、構造を誤認識する可能性が高くなる。更に、思考の中断が少ないために、筆跡が変形する場合は多いと考えられる。

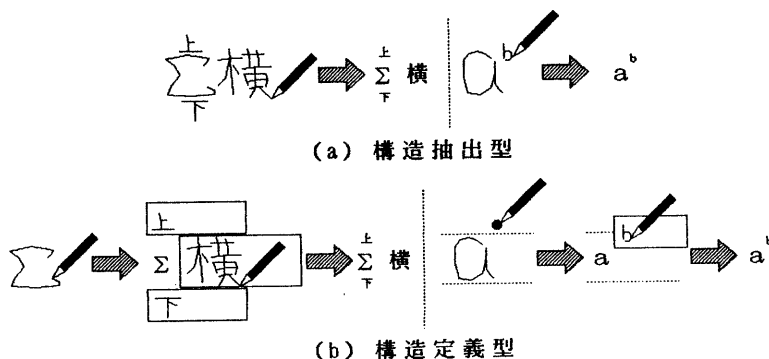


図1 構造抽出型と構造定義型

一方、構造定義型は、構造抽出型と逆で、ユーザが各部分の文字を書く前に構造を先に定義して、それから各部分の内容を書いてもらうという方式である。例えば、ユーザが“Σ”を記入した時点で、もしくはメニューから“シグマ”というボタンを選んで押したときに、“Σ”の下、上、横にそれぞれの部分の文字列を書き込むための箱を表示し、ユーザはその箱に文字を書いていくようにする。同様に、 a と書きたいければ、メニューから“上付き”というボタンを選んで押したり、あらかじめタブレット状に罫線を表示しておき、“a”が書かれている場所より上の部分をスタイルラスペンでポイントすると、右上に箱を表示し、そこに文字を書き入れるようにする（図1 (b) 参照）。この方法であれば、構造を誤認識することはなくなるが、思考の中断が生じてしまう。以上二つのタイプを要約すると次のようになる。

構造抽出型：思考の中断は少ないが、認識処理の負担は増す

思考の中断が少ないので、筆記は変形しがちになる

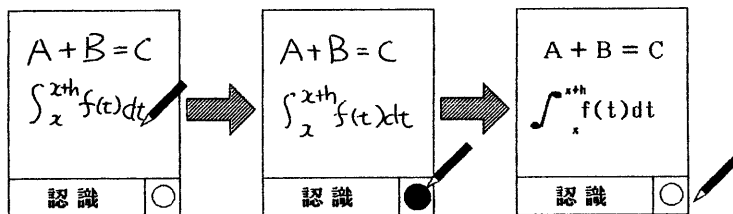
構造定義型：思考は中断されるが、認識処理の負担は減る

上で述べたものは、構造の作成方法の違いであったが、数式が入力されているときに、いつ認識系を通すか、つまり、認識のタイミングについても次に挙げる二つのタイプが考えられる。

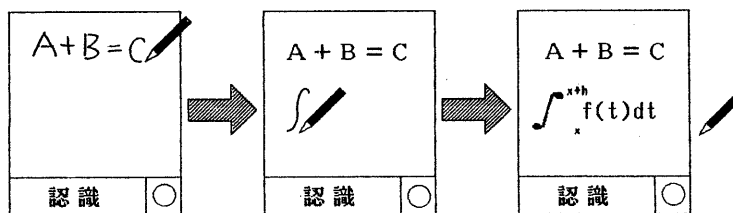
後処理型：すべての入力終了した後で、一括して認識する方法

逐次処理型：ある程度入力した時点で、逐次認識する方法

後処理型は、ユーザが計算機を使っていることを忘れて、あたかもペンで紙面上に数式を書いているように、つまり、思考の中断を起こさせないことをねらいとする方式である。具体的にいえば、ユーザが今ある数式をこのシステム上で解きたいとする。そのときには、その式の最初の状態から、途中経過、そして答まで、すべてを書き終わった時点で認識ボタンを押して認識をする（図2 (a) 参照）。しかし、この方法は、当然誤認識が多くなると考えられるので、入力終了後に大量の誤認識、リジェクトの修正をまとめてする必要がある。これに対し逐次処理型は、誤認識、リジェクトとその修正に伴う思考の中断に関してはこだわらないが、筆記と同時に計算機入力が進んでいくことを望むユーザ向けである。これは、ある程度数式が入力された時点で、つ



(a) 後処理型



(b) 逐次処理型

図2 後処理型と逐次処理型

まり、システムが数式の区切りを探し、その区切りの単位で認識をする方式である(図2(b)参照)。したがって、当然認識処理が入るたびに、誤認識の処理などのために思考の中断が生じる可能性が高い。以上二つの形態を要約すると次のようになる。

後処理型 : 入力時に思考は中断されないが、筆記後、誤認識、リジェクトなどの修正をまとめて行わなければならない

逐次処理型 : 入力中に誤認識、リジェクトが起こり、その修正で思考の中断が生じる

3. 3 構造化処理方法の組合せに関する考察

3. 2で、構造の作成方法と認識のタイミングに関して二つずつタイプを挙げたが、実際にシステムとして用いるときは、これらが組合さったものになる。そこで、次にそれぞれの組合せの特徴について考えてみる。

(1) 構造抽出型×後処理型

ユーザが区切りと考えるとところまで自由に書くことができるので、思考の中断が少ない。しかし、認識後の訂正には、多くの手間を要する。

(2) 構造抽出型×逐次処理型

認識する量が少ないので、前に誤認識した結果が後々影響することは少なくなるが、逐次認識処理が入ることにより思考の中断が生じ、加えて、文字の誤認識だけでなく、構造の誤認識も起こり得る。

(3) 構造定義型×後処理型

構造の誤認識はないが、思考が中断され、文字認識の訂正を入力後に最初から行わなければならない。

(4) 構造定義型×逐次処理型

思考は中断されるが、認識系の負担は減り、構造の訂正はない。文字の訂正に関しても前に入力したものに返ることがなく、訂正の手間が少なく済む。

我々は本システムの使い方として、ユーザの好み、または目的により2種類に分けられると考えている。ひとつは、とにかく計算機を使っているという感覚をなるべく持ちたくないもの。もうひとつは、多少時間がかかったり、思考の中断が生じることは構わないから、確実に仕事を行いたいというものである。前者を創造系、後者を複写系と呼ぶ。具体的には、創造系は、数式を実際にこのシステム上で解くような場合で、これは上に挙げた(1)の組合せが適していると考えられる。上記(1)の場合、筆記後に訂正を必要とするが、我々は、むしろこれらの作業は分離した方がよいとの仮説に立っている。一方、複写系は、例えば事前にユーザが解いたものや参考書などに載っている数式を綺麗な印刷物にしたい場合などで、これは上に挙げた(4)の組合せが適していると考えられる。筆者は、実際に両方の状態を満たすことができるようなシステムの構築を目指している。組合せ(2)、(3)は、お互いの長所をキャンセルしてしまうので、意味のある組合せとはいえない。

4. 文字認識部に関する実験

4. 1 対象文字セットとその特徴

現在ではまだ暫定的ではあるが、本システムで対象とする文字セットを、図3に示す。合計175文字になる。これは、日本語の文字認識システムの辞書に比べ、非常にコンパクトなものとなる。つまり、辞書を作成する労力もそれほど大きくなりということになる。したがって、辞書の仕様を変えたいとき、個人用の辞書を作りたいとき、ひとつの文字に対し複数のボタンを登録したいとき、などいろいろな要求に対応することが容易である。

ところで、数式に用いる文字パタンの性質とはいったい
 どのようなものだろうか。上で定義した文字セットを見て
 みると、最大で4画であることがわかる。つまり、数式で
 用いる文字パタンの性質の一つとしては、それらがいわゆる
 低画数文字であることがいえる。したがって、数式認識
 における文字認識処理では、低画数文字に対し認識率のよ
 い処理を用いなければならない。

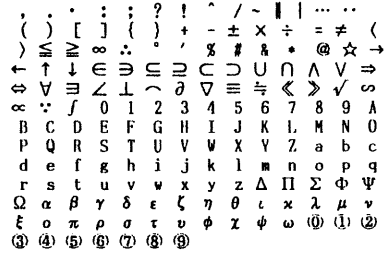


図3 文字セット

4. 2 低画数文字パタンの抱えている問題点

ここでは、低画数文字パターンに適した処理方法を見いだ
 すために、実際に高画数文字と低画数文字とでは、その特徴としてどのような違いがあるのかと
 いうことについて調べてみる。

まず、仮に各ストロークの持っている情報量が等しいと考えると、高画数になるほど情報量は
 増え、逆に低画数では情報量が少なくなってしまうということになる。しかし、はたしてこれが
 人間にも当てはまるだろうか。実際には、人間は必ずしも高画数に比べ、低画数の文字のほうが
 読み間違え易いということはないと思われる。では、どのようにして低画数の文字を区別してい
 くのだろうか。この疑問に対する答として二つのものが考えられる。

まず、一つ目には、高画数文字パターンに比べ、低画数文字パタンのほうが各ストロークの持つ
 ている情報量が高いのではないかというものである。つまり、低画数文字パターンでは、ストローク
 の向きや長さの微妙な差により、文字の識別を行っており、逆に、高画数文字パターンでは、細
 かな情報は、それほど文字を識別する要因とならないのではないだろうか。したがって、高画数
 文字パタンの認識においては、各ストロークから多少の情報をこぼしてしまっても認識すること
 ができるが、低画数文字パタンの認識では、より細かな情報を用いて認識することが重要と考え
 られる。

もう一つには、高画数文字は、1文字単独での認識の可能性が高いのに対し、低画数文字は、
 前後の文字の情報を用いた認識の必要性が高いのではないかというものである。概して、高画数
 文字パターンとしては、漢字が挙げられ、低画数文字としては、平仮名、片仮名、アルファベット
 が挙げられる。前者は、その文字自身に多くの意味を持つ文字種であるのに対し、後者は、それ
 ぞれの文字にはあまり意味のないものが多い。つまり、後者は、他の文字との組合せで意味のあ
 るものとなる場合が多い。したがって、人間が低画数文字パターンを認識するときには、その文字
 の前後にある文字の情報を多分に用いている可能性が高い。そこで、計算機における文字認識に
 おいても、文脈情報を用いた認識処理を用いることが、認識率の向上につながると考えられる。

次に、高画数文字パターンと低画数文字パタンの文字認識における特徴を簡単にまとめてみる。

<高画数文字パターン>

各ストロークの情報をそれほど細かく抽出しなくとも、各ストロークのおおまかな向き
 や、ストロークどうしの位置関係などに注目することにより、認識を行うことができる。

<低画数文字パターン>

ストロークどうしの位置関係などによる情報量だけでなく、各ストロークの持つ細かな情
 報や、前後の文字から得る文脈情報を用いて認識しなければならない。

筆者は、上で挙げたもののうち、各ストロークのより細かな情報を抽出することにより認識率
 を向上させるという方法に着目し、実験を行った。

4. 3 低画数文字パタンの認識率を向上させるための実験

現在我々の研究室では、JOLIS-2Eというオンライン手書き文字認識システムが稼動中であ

る。基本的には、筆者が構築するシステムの文字認識部には、この JOLIS-2E を用いる。しかしながら、JOLIS-2E は日本語対応のシステムなので、これを数式専用に変更する必要がある。そこで、今回は上記 4. 2 で得たことをもとに、JOLIS-2E に変更を加え、その結果を見てみた。

JOLIS-2E では、あらかじめ基本ストロークというものを定義し、入力されたストロークをどれかの基本ストロークに対応させ、文字を基本ストロークの列で表現し、文字認識を行っている。したがって、入力された文字が低画数の場合、基本ストロークの識別結果が文字認識に大きな影響をおよぼす可能性があるわけである。つまり、数式専用にするためには、このストローク識別の部分の強化が必要であると考えられる。ストローク識別の中では、入力された筆点列をベクトル列に変換する処理がある。このとき入力された筆点の n 点目と $n+k$ 点目 (JOLIS-2E では $k=3$) を結んでできるベクトルをその方向により 8 種類に分け、ベクトルの記号列を作成する。しかし、この “3 点スパン (間隔)” と “8 方向分割” が上記のより細かな情報を抽出するということに適していないと考え、これを 1 点スパンと 16 方向分割に変更してみた。図 4 にスパンの変更結果、図 5 に分割の変更結果を示した。まず、スパンの変更についてだが、3 点スパンの方は「{」の中央の転回が削れているのに対し、1 点スパンでは転回が残っていることがわかる。次に、分割の変更についてだが、8 方向ではベクトル列が筆点列からかなり離れているのに対し、16 方向では非常に近くなっていることがわかる。

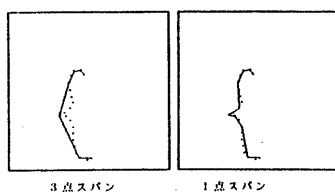


図4 スパンの変更結果

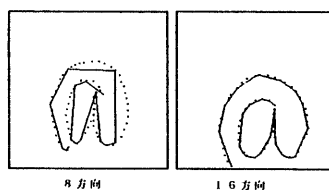


図5 分割の変更結果

5. おわりに

現在、JOLIS-2E に更に改良を加えている段階だが、数式認識における文字認識処理としては、

- ・認識系の組合せや順番
- ・意味的情報を用いた認識処理

について考えてみる必要がある。数式認識システム全体としては、

- ・画面上への結果出力
- ・他のアプリケーションとの関連性
- ・手書きを生かした訂正方法

などについてが今後の課題である。

参考文献

- [1] 曾谷俊男他：“手書きユーザインタフェース”，第30回プログラミングシンポジウム (1990)
- [2] 平松 徹他：“オンライン手書き文字認識システム JOLIS-2 の構成”，
情報処理学会第35回全国大会 3H-5 (1987)
- [3] 村瀬敦史他：“手書き数式入力におけるヒューマンインタフェースの考察と認識システムの
実現”，情報処理学会第42回全国大会 5D-7 (1990)
- [4] 宮崎 一哉：“ユーザインタフェースに対する要求—エンドユーザの立場から—”，
ヒューマンフレンドリーなシステムシンポジウム報告集 267-273 (1986)