データベース・システム 85－1
プログラミング－言語・基礎・実践－ 4－1
（1991. 10. 4）

ストロークデータのファイル構造と管理プログラム

森田利広　守屋慎次
東京電機大学工学部

ペン入力データとテキストデータとをMS-DOSパソコン上で扱うファイル構造と，このファイルをアプリケーションで扱うための管理プログラムとそのアーキテクチャを示す．本ファイル構造では，ストロークとテキストという異なるメディアに加えて多様な属性を一括して，またはそれぞれを個別に，入出力や更新ができる．これにより，異種メディアや多様な属性を扱う自由度が増し，アプリケーション間のファイル互換性が確保される．なお，本ファイル構造のデータ容量とデータ入出力時間について測定を行ない，十分実用に耐える結果を得た．

A FILE STRUCTURE FOR STROKE DATA AND ITS FILE MANAGEMENT PROGRAM

Toshihiro Morita　Shinji Moriya

Department of Electrical Communication Engineering, Tokyo Denki University

2-2 Nishiki-cho, Kanda, Chiyoda-ku, Tokyo 101, Japan

This paper proposes a file structure for pen-based stroke data (we call this file structure the Stroke File) and a management program developed using the Stroke File. The Stroke File consists of various kinds of data such as stroke and text and their attributes. The management program will help designers and programmers in developing programs which input/output using the Stroke File. We measured the file size and data input/output time for stroke data which used this file structure, and we conclude that this file structure can be used practically.

1．まえがき

近年，ペンを入力装置として用いるタブレット形のコンピュータ（以後ペン入力コンピュータ）が数多く出現している．これらの装置で扱われる主なデータは，手書き入力などによるペン先の位置を示すxy座標値の時系列である．これをストロークデータと呼ぶ．

本論文の目的は，パソコン上でストロークデータとテキストデータとを扱うファイル，すなわちストロークファイルの構造を示すことと，その管理プログラムを作成することである．

ペン入力コンピュータとして，SONYのPalmTop[1]やGRiD SYSTEMSのGRiDPAD[2]などが，またそのオペレーティングシステムとしてMicrosoftのWindows for Pens[3]やGO Corp.のPenPoint[4]が実用化されつつある．だが，これらの各システムにおけるファイルはそれぞれ独自の形式で格納されている．今後，ペン入力コンピュータが普及するための条件として，ファイル中におけるストロークデータの構造や扱い方を研究し定めていくことは重要である．

本ファイル構造では，ストロークデータとテキストデータという異種メディア，およびそれらの属性が1つのMS-DOSファイルに格納される．このため，テキストを扱うアプリケーションプログラム（以後アプリケーション），ストロークを扱うアプリケーション，テキストとストロークの両者を扱うアプリケーションの，それぞれの間でファイル互換性が確保できる．また，管理プログラムを用いることにより，ストロークファイルを入出力するアプリケーションの作成が容易になる．このため開発効率と信頼性の向上，保守の簡易化，プログラム仕様の標準化を図ることができる．

本論文では，まず2．でストロークファイルの構造に対する要求を挙げ，3．で具体的なストロークファイルの構造，4．で管理プログラムとその利用例について述べる．5．ではストロークファイルと管理プログラムの特性について述べる．

2．要求されることがらと扱う情報

本章では，ストロークファイルとその管理プログラムに対して要求されることがらを，アプリケーションの立場から列挙する．次に，ストロークファイルで取り扱う情報の種類と構造について述べる．

2.1 ファイル構造に対する要求

筆者らは，以前よりペン入力データを扱うアプリケーション（以後ペン入力アプリケーション）をいくつか開発済，あるいは開発中である．それらのうちここでは，「切出しシステム」[5]，「室内会議システム」，「電筆システム」[6]の3つをあげ，各システムからのファイル構造に対する要求を述べる．また，システム間のファイル互換性や，システム開発者からの要求についても述べる．

(1) 切出しシステム[5]における要求

用紙1枚分の手書き文章をストロークデータとして採取し，それを行，字などの単位で切り出すアルゴリズムが筆者らのグループにより研究されている[5][7][8]．切出しシステムは，手書き文章データの採取と，各種の切出し実験とを行うものである．

まず，用紙が表示されているタブレット画面上で自由に文章を筆記し，これをストロークデータとして採取する．ここでタブレット上で動くペン先の軌跡に注目し，これを「実画」と「虚画」の繰り返しと考える．実画とはペン先が画面上を滑った部分，虚画とはペン先が空中を移動した部分である．図1に，手書き文章における実画と虚画の実例を示す．採取されるデータは，一定の時間間隔（例えば0.01秒）ごとにサンプル入力されるxy座標値（以後，点と呼ぶ）の時系列である．これを図2に示す．

採取されたストロークデータはファイルに格納される．次に，採取されたストロークデータに対して字や行の切出しを行う．切出しの結果，字の切れ目，行の切れ目といった印が虚画に付与される．この結果は画

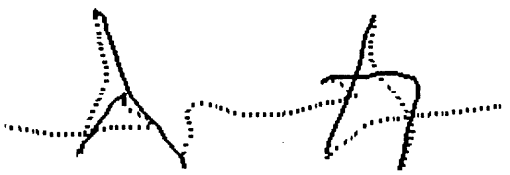

図1　ストロークデータの実例．実線は実画，破線は虚画をそれぞれ示す．

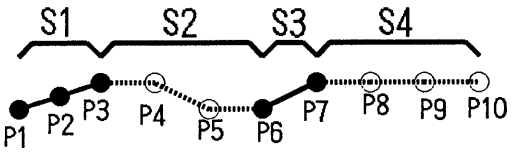

図2　ストロークデータの例．黒丸と白丸は，タブレットから計測された点データを示す．P1～P10の順に計測されたとする．黒丸と白丸はそれぞれ実画(S1,S3)と虚画(S2,S4)中の点を示す．実画は実線で，虚画は破線で結んで示した．

面に表示されるほか，元のファイルにも格納される．

切出しシステムにおける要求を次に示す．

A1　切出しシステムでは，実画と虚画の両者を扱う．一方，後述する室内会議システムでは実画のみを扱う．よって，各システムで共通に扱えるファイル構造とするために，管理プログラムは実画と虚画のファイル入出力，実画のみのファイル入出力，虚画のみのファイル入出力のいずれも行える必要がある．

A2　特定の個人が筆記したデータや，特定の用紙画面上に筆記されたデータについて実験を行い，個人ごとや用紙ごとの正切出し率を求めたいことがある．このため，筆記者名や用紙の種類など，筆記時の様々な条件をストロークデータとともにファイルに格納したい．

A3　字や行の切れ目を示す情報をストロークデータとともにファイルに格納したい．

(2) 室内会議システムにおける要求

室内会議システムは，入出力一体型タブレットを用いてテキストデータ上に手書き入力を行い，その画面をOHPに映し出すものである．主に，少人数による非公式な会合やブレーンストーミングにおける討議用の資料作成に用いることを目指している．

1つの画面（スライドやOHPシートに相当する）には，ストロークデータ（手書き）とテキストデータを重畳して表示できる．また，手書きやテキストの一部を強調，反転，点滅させるなどの飾り付けも行える．完成した画面はファイルに保存される．会合時には，画面上に説明用の手書きを加えることもできる．

室内会議システムにおける要求を次に示す．

B1　ストロークデータとテキストデータの両者を一括して，また，それぞれを個別にファイルに格納したい．

B2　ストロークデータ，テキストデータのそれぞれに対する飾り付けの情報をファイルに格納したい．

(3) 電筆システム[6]における要求

二者のそれぞれが入出力一体型タブレットを用い通信回線を経由して筆跡（ストロークデータ）を送受しあい，1対1の筆談を行うシステムである．

まず筆談を開始する前に，あらかじめ用件を記入した画面を作成しておく．作成した画面は一旦ファイルに格納される．筆談相手との通信回線が接続されると，あらかじめ格納されたファイルの内容が一括して相手側へ転送できる．

次に，実時間で筆跡を送受しあい，画面上で筆談を行う．筆談した内容は必要に応じて各個人のファイルに保存され，後で内容を見たり編集することもできる．このときあらかじめ書かれていた部分と追記した部分とを区別して表示できる．

電筆システムにおける要求を次に示す．

C1　データの転送時間を短くするため，ストロークデータのファイル容量は小さく実現したい．

C2　筆記された画単位で，追記か否かを区別するための情報をファイルに保存したい．

(4) システム開発者からの要求

システム開発者からの要求を以下に挙げる．

D1　通常，プログラミング言語はテキストデータの入出力命令を備えている．ストロークデータの入出力も，テキストデータの場合と同様なプログラミングスタイルで実現できれば，プログラム作成は容易になる．

D2　多くのプログラミング言語におけるテキストデータの入出力は，字単位，行単位のいずれでも行うことができる．ストロークデータの入出力も点単位，画単位のいずれでも行えると自由度が増す．

D3　スタイラスペンには，ペン先スイッチのon/offとxy座標値とが採取できる通常のペンと，スイッチはついていないがペン先に加わる圧力とxy座標値とが採取できる筆圧ペンとがある．通常のペンと筆圧ペンを筆記中に交互に用いることも考えられるが，この場合，画単位でペンの種類を表す情報をファイルに格納したい．また，筆圧ペンで筆記した画については筆圧のデータをファイルにも格納したい．

2.2 扱う情報の種類と木構造

前節の要求をもとに，ストロークファイルに格納するデータの分類を木構造で表した．これを図3に示す．木構造の根(stroke file)は1つのストロークファイルを表している．葉（①～⑱）はストロークファイル中に格納されるデータを表す．①～⑱の並びがそのままストロークファイルの構造となっている．ファイル構造の詳細については次章で述べる．

図2左側の枝と節はメンバの階層的分類を意味する．分類の基準は「メディア」，「実と虚」，「データと属性」の3つからなる．以下でそれぞれについて述べる．

(1) メディア

本ファイル構造では，要求B1で述べたようにストローク(stroke)とテキスト(text)の2種類のメディアを扱うこととする．

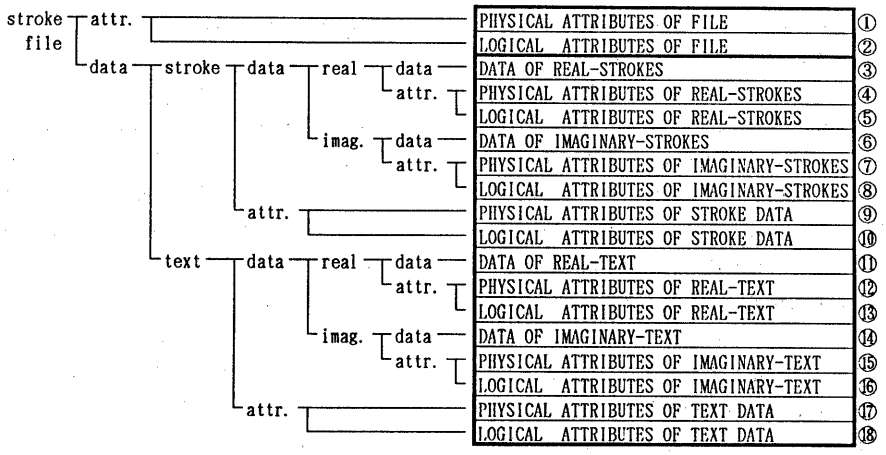

図3　ストロークファイル全体の構成（右側の枠内）と構成要素の階層構造（左側の木）．'attr.'は'attributes'の，'imag.'は'imaginary'のそれぞれ省略形を示す．枠内の大文字の部分はその部分がより細かな要素から構成されていることを示す．

(2) 実データと虚データ

2.1節(1)でストロークデータの実画と虚画について述べた．これを一般化して，実データ(real data)と虚データ(imaginary data)を次のように定義する．

・実データ…入力者が入力したかったデータ

・虚データ…実データを入力するときに必要となった入力者の動作の一部を記録しているデータ

一例として，テキストの実データと虚データを考えてみる．テキストファイルにおける実データとは，入力者がキーボードを操作し，文書ファイルとして実際に入力されたアルファベットや記号などの文字列である．一方，虚データは文字列を入力するために必要となったカーソル移動やウィンドウ操作やメニュー操作など，入力者の動作のうち，コンピュータ内に入力される部分といえる．

同様に，他のメディアでも人がデータの入力を行うので，実データと虚データが存在しうる．そこで，本ファイル構造では各メディアにおいて実データと虚データとが扱える構造とした．

(3) データと属性

要求Ａ３，Ｂ２，Ｃ２で述べた切れ目や飾り付けや追記の情報は属性と呼んでよい．また，要求Ａ２で述べた用紙の種類などもストロークファイル全体に対して付与される属性と呼んでよい．

属性は，その内容により物理属性(physical attribute)と論理属性(logical attribute)に分けられる．物理属性はハードウェアにより定められた仕様を表すもので，ペンの種類やタブレットの種類などがこれに相当する．論理属性はアプリケーションの使用目的に合わせて定義されるもので，画の太さや表示色，手書き文章における字や行の切れ目などがある．本ファイル構造では物理属性と論理属性の両者を扱う．

3．ストロークファイルの構造

本章では，ストロークファイルの具体的な構造について，２章の各要求に応えながら述べる．

3.1 ファイル全体の構成

本節ではまず，ストロークファイルの全体的な構成について述べる．

２章で述べた要求のうち，ファイル構造に対するものを整理すると次の点にまとめられる．

・図３の①～⑱を，個別または一括して，MS-DOSファイルとして入出力し更新できるようにしたい．

上に挙げた点を満たすため，ストロークファイルは区分ファイル編成[9]を用いて実現した．

１つの区分ファイルは，複数のメンバ[9]とそれらを管理する１つのディレクトリから構成される．①～⑱のそれぞれが１つのメンバとして保存される．メンバ内のデータは順アクセスすることができる．ディレクトリは，ファイル中における各メンバの開始位置を保持する．ファイル中の各メンバは別々に入出力できる．このため，要求Ｂ１で述べたストロークデータとテキストデータの一括または個別の入出力や，要求Ａ１で述べた実画と虚画の一括または個別の入出力が可能と

なる．

ストロークファイルは区分ファイルであるが，図3に示した階層的分類をもとにして構成されている．①から⑱のうち，②には区分ファイルのディレクトリが格納される．区分ファイルにおいてはディレクトリの位置は固定されなくてはならないため，①，②はファイルの先頭に位置する．一方③～⑱は必ずしも図2に示した順序に並ぶ必要はない．また，③～⑱の任意のものを省略したり，③～⑱のうちの任意のいくつかをそれぞれ複数置くことも可能である．例えば，入力時点における筆跡と，何らかの処理後の筆跡の両者を一つのファイルに保存できる．

3.2 メンバの構造

3.1節で，図3の①～⑱のそれぞれが1つのメンバとして保存されることを述べた．そこで①～⑱のそれぞれを以後はメンバと呼ぶことにする．本節では，主要なメンバの構造について述べる．

図2のようなストロークデータがストロークファイルに格納された場合の，各メンバの構造と内容例を図4に示す．以下で，それぞれについて説明する．

(1) ファイルの論理属性（メンバ②）

区分ファイルにおけるディレクトリを保持する．構造を図4(a)に示す．メンバの識別子(member id)は，メンバの種類ごとに管理プログラムが割り当てた数値である．開始アドレス(address)は，ストロークファイル中におけるメンバの開始位置を表す．ファイル中に存在する各メンバに対する識別子と開始アドレスとが，メンバが格納されている順に書き込まれる．最後の end of member はメンバ②の終了を表す．

ディレクトリを読みだしたり更新したりする操作は，すべて管理プログラムが自動的に行う．このため，管理プログラムの利用者はディレクトリの存在を意識する必要がない．

(2) 実ストロークデータ（メンバ③）

ストロークデータのうち実画のデータを保持する．構造と内容例を図4(b)に示す．メンバ⑥も同様の構造である．

要求Ｄ２より，ストロークデータの構造は，テキストファイルの構造と考え方を次のように合わせた．テキストファイルの最小単位は文字であり，文字の並びとして行が，行の並びとしてファイルが構成される．また，行の終わりやテキストの終わりを表すために，制御文字が設けられている．

ストロークデータの最小単位は点である．そこでまず，数バイトで1つの点を表し，これを点レコード(point)と呼ぶ．次に，点レコードの1次元的な並びとして1つの画が，画の並びでストロークデータが構成される．また，画の終わり(end of stroke)やストロークデータの終わり(end of member)を意味する制御点が

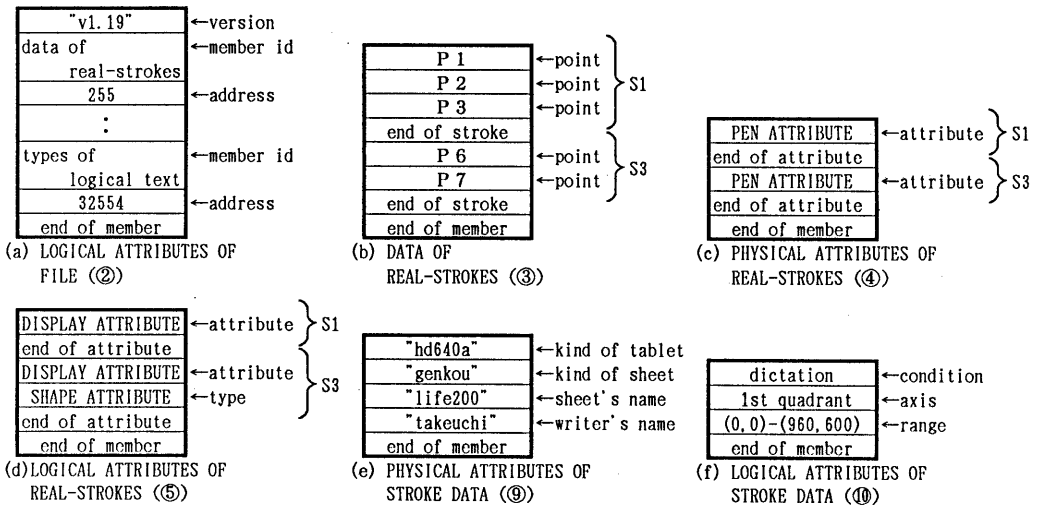

図4　(a)～(f)の枠内はそれぞれ，図2に示した②③④⑤⑨⑩のより詳細な構造と内容例を示す．枠内の大文字の部分はより細かな要素から構成されていることを示す．枠内の" "で囲まれた文字列はその文字列がその場所に，小文字の部分にはそれに対応する符号がその場所に，それぞれ格納される．

設けられている．このように，テキストファイルと同等の考え方でストロークデータのメンバを構成した．

次に，図4(b)中の実画Ｓ１を構成する三つの点レコードＰ１，Ｐ２，Ｐ３のそれぞれの構造を図５に示す．Ｐ１は絶対座標点(absolute)，Ｐ２，Ｐ３は相対座標点(relative)で表されている．絶対座標点は，タブレットが計測するxy座標値をそのままレコード中に格納している．相対座標点は，直前点からの相対距離によりxy座標値を表す．相対座標点を用いて座標値を相対的に表すことにより，絶対座標点のみを用いた場合の約3/5程度にファイル容量を縮少できる．これは要求Ｃ１に応えるものである．

点レコード中のxy座標値は，タブレットのもつ座標軸の向き（例えば第１象限）や原点の位置，空間分解能に依存する．これらを座標系データと呼ぶことにする．一般には，アプリケーションの種類により扱う座標系は異なる．そこで，ストロークデータと共に座標系データもファイルに格納することとした．これは以下に述べる(4)ストロークの論理属性で扱われる．

(3) 実ストロークの物理(論理)属性（メンバ④，⑤）

画単位で付与される属性を保持する．メンバの構造を図4(c),(d)と図６に示す．メンバ⑦，⑧も同様の構造である．

属性からなるレコード(attribute)には，画の太さや表示色，字や行の切れ目などの属性を格納する．属性の並びとそれに続く制御情報(end of attribute)により，１画に付与される属性が表される．主な属性の種類を以下に示す．

・ペンの種類（PEN ATTRIBUTE） …スタイラスペン

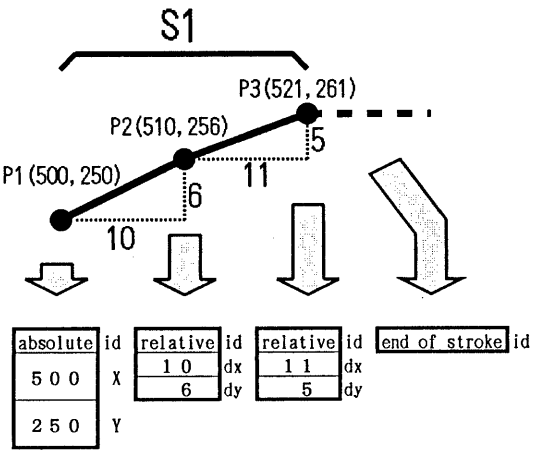

図5　図4(b)に示した点レコードP1, P2, P3の詳細な構造と内容例

には，筆圧を採取するものとしないものとがある．筆圧ペンで筆記された画については，本属性が付与されるとともに筆圧のデータが点レコードに付加される．これにより要求Ｄ３で述べた筆圧ペンと通常のペンとを混合して使用することが可能となる．

・画の形状（SHAPE ATTRIBUTE） …画の形状を表す．具体的には，直線，左曲がりの曲線，右曲がりの曲線，曲線中のループの有無などの情報が格納される．

・画の表示属性（DISPLAY ATTRIBUTE） …画の線種（実線，点線など），線の太さ，表示色などを表す．

(4) ストロークの物理(論理)属性（メンバ⑨，⑩）

手書き文章の筆記時におけるタブレットの種類などは，筆記中に変化することはない．この属性は，画ごとではなくストロークデータ全体に対して付与される性質のものである．このような属性をストロークの物理属性（図4(e)），ストロークの論理属性（図4(f)）として扱う．これらの属性は，必ずしもメンバとして扱うほどのデータ量ではないが，ここでは他のデータや属性と同様な，一貫したプログラミングスタイルで実現するために，メンバとして表すこととした．

以下で，各項目について説明を加える．

・タブレットの種類（kind of tablet） …筆記に用いたタブレットの種類を表す．

・用紙の種類(kind of sheet)と用紙名(sheet's name) …用紙とは，入出力一体型タブレットにおいては液晶画面上に表示されている罫線，入力専用タブレットにおいてはタブレット上に貼り付けた実際の用紙を表す．用紙の種類は白紙，原稿用紙などの種別，用紙名は用紙固有の名前をそれぞれ表す．

・筆記者名（writer's name） …筆記者の名前を文字列で表す．

・筆法（condition） …考えながらの筆記，転記など，どのような状態で筆記が行なわれたかを表す．

・座標軸の向き（axis） …xy座標軸の向きを表す．

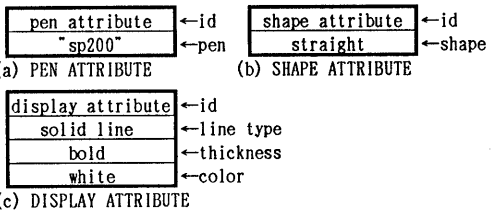

図6　図4の(c)(d)中に大文字で示した部分（属性のレコードを示す）の詳細な構造と内容例

・座標値の最大値／最小値（range） …ストロークデータ中における，x座標値，y座標値それぞれの最大値と最小値を表す．座標平面上のどの範囲にストロークデータが存在するかを示す．

(5) テキストを扱うメンバの構造

図２中⑪～⑱のメンバは，テキストを扱うメンバである．このうち，実（虚）テキストデータ（メンバ⑪，⑭）の構造はMS-DOSテキストファイルと全く同じ構造となっている．メンバ⑫，⑬，⑮～⑱は，(2)～(4)で述べたストロークを扱うメンバとほぼ同様の構造となっている．本論文では，これらのメンバの構造については省略する．

4．ファイル管理プログラム

本章では，ストロークファイルの入出力を行う管理プログラムについて述べる．

4.1 ファイル管理プログラムの構造

本節では，ファイル管理プログラムの具体的な構造について述べる．

図７は，ストロークファイルとファイル管理プログラムまわりの説明図である．図７は以下の４つの部分に大別される．

・ペン入力アプリケーション （pen-based application） …ペン入力データを扱うアプリケーションプログラムである．ファイル管理プログラムで定義された手続き・関数を用いることにより，ストロークファイルの入出力を行うプログラムが作成される．

ストロークファイルを入出力する手続き・関数の一覧を付録Ａに示す．付録中の「メンバ」は図３で示したメンバの番号に対応する．このように，各メンバごとに入出力用の手続き・関数が用意されている．プログラム言語としてはTurbo Pascal 5.5[10]を用いている．この理由は，プログラムが読みやすいこと，オブジェクト指向[11]の採用によりデータ構造の隠蔽がしやすいこと，２章で紹介した各応用プログラムが本言語で記述されていること，による．

・内部データモデル （internal data model） …主記憶装置におけるストロークデータの構造である．画のリスト構造と点のリスト構造が階層的に連なった構造となっている．詳細は文献 (12)を参照されたい．内部データモデル中のストロークデータは，常に付録Ａに示した手続き・関数を呼び出すことにより入出力される．このため，ペン入力アプリケーション側からみると内部データモデルの構造は隠蔽される．

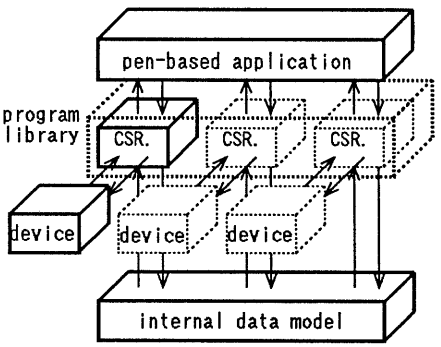

図7　ストロークファイルと管理プログラムまわりのアーキテクチャ．CSR.は機能カーソルを，deviceは仮想入出力装置をそれぞれ表す．破線で示したCSR.とdeviceは，現在作成中であることを示す．

・操作ライブラリ （program library） …内部データモデルの生成，編集，内容の取り出しを行うプログラムライブラリである．図７中の「CSR.」は，機能カーソルと呼ばれるオブジェクト(object)[11]である．ここでオブジェクトとは，一群のデータとそれを操作する手続き・関数をひとまとめにカプセル化したものを指す．機能カーソルはその名前が示すように，内部データモデル上をあたかも画面上のカーソルのように動きまわりながら，内部データモデル中のxy座標値を特定の仮想入出力装置（後述する）やペン入力アプリケーションに対して入出力する．また，内部データモデルの生成，編集などの操作を行う．

ファイル管理プログラムは，ファイル入出力を行うための機能カーソルを持っており，これをファイルカーソル(FileCursorCell)と呼ぶ．ファイルカーソルは，ストロークファイルの内容を内部データモデルに対して入出力する．なお，ファイルカーソルを用いずに，ストロークファイルの内容を直接読み書きする手続き・関数もある（ストロークの読み書きの場合だけを例示すると，付録ＡのWritePosとReadPosがこれに相当する）．

・仮想入出力装置 （device） …ペン入力アプリケーションが扱う仮想的な装置である．ファイル管理プログラムではストロークファイル中の特定の１バイトを指すファイルポインタ(StrkFile)が定義されており，ファイルカーソルはこのファイルポインタを仮想的な装置と見なしてストロークファイルへのアクセスを行う．他のハードウェア（タブレット，表示画面，プリンタなど）における仮想装置についても，ファイルカ

ーソルと同様の考え方によって操作ライブラリを作成中である．

4.2 ファイル管理プログラムの利用例

本節では，ファイル管理プログラムを用いたプログラム例を示す．

ストロークファイル中の実ストロークデータを内部データモデルに読み込む例を図８に示す．以下で，プログラムの内容について説明する．

１行～３行は変数宣言である．プログラム中で用いている変数は，前節で述べたファイルポインタfp，ファイルカーソルfcs，一時変数foundの３つである．

４行のSeekFirstMbrは，ファイル中から任意のメンバを探し出す手続きである．ここでは，引数としてDataOfRealStroke（実ストロークデータ）を指定している．メンバが見つかったか否かは，一時変数foundにより分かるが，図８ではその判定を省略した．

５行～11行のwhileループで，ファイル中にある全ての実画を内部データモデルに読み込んでいる．関数EOWrtにより end of member を検出する．

６行～８行のwhileループでは，ファイル中の１画を読み込んでいる．EOStrkにより end of stroke を検出する．

７行のReadPntで１点を読み込む．これはPascalのRead手続きに相当する．読み込んだ点は１画単位でファイルカーソル中のバッファに蓄えられる．バッファ中のデータは，10行のNewTailStrkにより内部データモデル中にある画のリストの末尾につながれる．

```
 1 var fp    : StrkFile ;
 2     fcs   : FileCursorCell ;
 3     found : Boolean ;
   ...
 4 SeekFirstMbr( fp, DataOfRealStroke, found );
 5 while not EOWrt( fp ) do begin
 6     while not EOStrk( fp ) do begin
 7         fcs.ReadPnt( fp ) ;
 8     end ;
 9     ReadEOStrk( fp ) ;
10     fcs.NewTailStrk( New(
         StrkLink, Init( RealStrk )));
11 end ;
12 ReadEOWrt( fp ) ;

        fcs.ReadWrt( fp );

        fcs.ReadTailStrk( New(
        StrkLink, Init( RealStrk )));
```

図８　管理プログラムを用いてストロークデータを内部モデルへ入力する例．太字は管理プログラムの手続き・関数，他の記法は Turbo Pascal の記法に準ずる．

９行のReadEOStrk，12行のReadEOWrtはそれぞれ end of stroke，end of memberを読み飛ばす．これはPascalのReadln手続きに相当する．

図８のプログラムは点単位で入力を行っているが，内側の点線部分を図８の下側に示したReadTailStrkで置き換えて画単位で入力を行ったり，外側の点線部分をReadWrtで置き換えてストロークデータ全体を一括して入力したりできる．これは要求Ｄ２に応えるものである．

５．データ容量と入出力速度の特性

本章では，ストロークファイル構造と管理プログラムを用いた場合のデータ容量，入出力時間の特性を測定した結果について述べる．

5.1 実験対象データ

測定に用いたデータは，図１に示した漢字，仮名混じりの日本語文章である．市販のB5判，200字詰めの横書き原稿用紙をタブレット上に置き，この上で自由に文章の筆記を行った．用紙上の１マスの大きさは縦10mm×横10mmである．使用したタブレットのサンプリング速度は100[点/sec]，空間分解能は20[点/mm]である．なお，採取したストロークデータに対して，点の間引きなどの処理は一切行なっていない．

以上の条件で，原稿用紙１枚上に195字分の実画および虚画のデータが採取された．

次に，切出しシステム[8]の字切出しアルゴリズム[8]を用いて，ストロークデータを１字ごとに切り出した．字切出しが済んだ手書きデータから，5字分，10字分，15字分，…，195字分のそれぞれについて，実画と虚画とからなるファイル39個，および実画のみのファイル39個を作成した．これらのファイルを対象として，以下に述べる実験を行った．

5.2 データ容量の測定

前節の方法で作成した合計78個のファイルについて，ファイル容量および内部データモデル中のメモリ容量を測定した．結果を図９に示す．

実画のみの場合，ファイル容量は195字で約29Kバイト，内部データモデルの容量は約96Kバイトとなった．ファイル容量は内部データモデルの容量の1/3以下となっている．この原因としては，主記憶中では線形リストを用いている[12]ためポインタの容量が大きいこと，一方ファイル中では図５で示した相対座標点を用いて容量を減らしていることが考えられる．

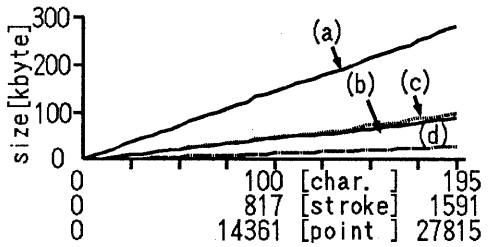

図9　ストロークファイルのデータ容量．char.，stoke，point はそれぞれ字，画，点の数を意味する．（画と点の数は (a)(b)の場合を示す）
(a)実画と虚画のﾒﾓﾘ容量　(b)実画と虚画のﾌｧｲﾙ容量
(c)実画だけのﾒﾓﾘ容量　(d)実画だけのﾌｧｲﾙ容量

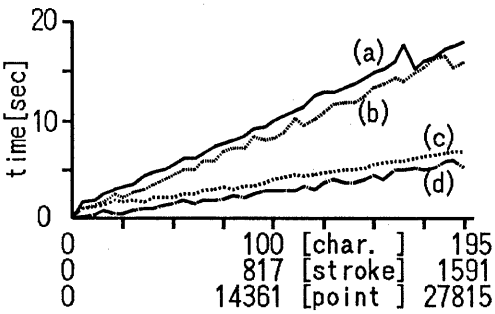

図10 ストロークファイルの入出力時間．char.，stoke，point はそれぞれ字，画，点の数を意味する．（画と点の数は (a)(b)の場合を示す）
(a)実画と虚画の出力時間 (b)実画と虚画の入力時間
(c)実画だけの出力時間　(d)実画だけの入力時間

5.3 ファイル入出力時間の測定

5.1節で作成した78個のファイルを空のフロッピディスクに書き込む時間と，書き込んだデータを再び読み込む時間を測定した．使用したパソコンはNEC㈱製PC-9801RA21で，クロックは20MHzである．

測定結果を図10に示す．書き込み，読み込み時間ともに字数にほぼ比例しているが，全体的に不安定さが目立つ．これはディスクヘッドの移動の仕方などが原因と考えられる．また，全体的に書き込み時間の方が読み込み時間よりも長い．これについては，ファイル編成上の原因が考えられる．すなわち，ファイル書き込み時は，必要なメンバをすべて書き込んだ後，ファイルクローズ時にディレクトリを更新しなくてはならず，このためヘッドを移動する時間とディレクトリを書き込む時間が加わっていると思われる．

また，実画のみの読み込み，書き込み時間は195字で約7秒となった．

6．むすび

本論文では，ストロークとテキストとを扱うファイル構造と，その管理プログラムについて述べた．

本ファイル構造は，2章で述べた3つのアプリケーションと，筆者らが開発中の他のアプリケーション上で実用されており，特にファイル互換性の点で効果をあげている．

ファイル構造を決定するにあたって，まずストロークデータを扱う複数のペン入力アプリケーションを分析し，ファイル構造と扱う情報に対する要求を把握した．次に，この要求をもとに，ファイルに保存する情報の種類を系統的に整理し，ストロークとテキスト，実と虚，データと属性という分類を示した．さらにその結果，区分ファイル編成を用いてストロークファイルを実現した．

また，ストロークデータの入出力をテキストデータの入出力と同様なプログラミングスタイルで実現できるようにした．ファイルの入出力は機能カーソルにより行われ，実際のファイル構造や主記憶中のデータ構造は隠蔽される．このアーキテクチャは，ディスクドライブ以外の入出力装置とも同様な考え方とした．

また，実際のファイル入出力に要する時間や，データ容量について測定を行った．ストロークデータのファイル入出力時間については，十分実用に耐えうるが，まだ満足できる水準に達していない．また，ファイルの容量はかなり大きくなっている．これらの問題点を解決するため，高速で効率のよいストロークデータの圧縮技術が必要となるであろう．

今後の課題としては，上記の特性を改善するとともに，本ファイル構造を発展させてデータベース化すること，文書交換用の標準規格に本ファイル構造を対応させること，が考えられる．

文　献

(1) PalmTop Computer PTC-500 解説書，ソニー(1990).
(2) “The Pen: Computing's Next Big Leap”，Business Week/May，Vol.14，pp.40-41(1990).
(3) “Windows for Pens Windowsアプリをペンで操作可能に”，日経バイト，No.90，pp.372-385(1991).
(4) Robert Carr，Dan Shafer: “The Power of PenPoint”，Addison-Wesley Publishing Company (1991).

(5) 守屋慎次，稲井幸治，田村岳史，清水　聡，S.N.ｸﾘｼｭﾅﾝ，“運筆データの採取と単位情報の認識”，計測自動制御学会 Human Interface N&R, Vol.4, pp. 268-278(1989).

(6) 守屋慎次，木村龍英，稲井幸治，檜垣誠一，谷中　大：“電筆と電話による実時間コミュニケーション”，計測自動制御学会第6回ヒューマン・インタフェース・シンポジウム論文集，pp.221-230 (1990).

(7) 守屋慎次，清水　聡，S.N.ｸﾘｼｭﾅﾝ，“運筆データからの行の切出し”，電子情報通信学会論文誌(D-II)，J73-D-II, 7, pp.973-981(1990).

(8) 森田利広，守屋慎次，清水　聡：“運筆データからの字の切出し”，電子情報通信学会論文誌(D-II)，J73-D-II, 10, pp. 1796-1798(1990).

(9) 山谷正己：“ファイル編成入門”，オーム社.

(10) Borland International: “Turbo Pascal 5.5 オブジェクト指向プログラミングガイド”，㈱ボーランドジャパン.

(11) Bertrand Meyer著，二木厚吉 訳，酒包　實，酒包順子 共訳：“オブジェクト指向入門”，アスキー出版局.

(12) 森田利広，守屋慎次：“ペン入力データの内部モデルとその操作ライブラリ”，計測自動制御学会 Human Interface N&R, Vol.6, pp. 261-268(1991).

付録A　ファイル管理プログラムの手続き・関数一覧

メンバ	手続き・関数名	機　能
	AssignStrkFile	ﾌｧｲﾙﾎﾟｲﾝﾀをファイル名と結び付ける
	RewriteStrkFile	ファイルの新規出力オープン
①	AppendStrkFile	ファイルの追加出力オープン
	ResetStrkFile	ファイルの入力オープン
	CloseStrkFile	ファイルのクローズ
	CompressStrkFile	ファイル中の不要メンバを圧縮
	SeekFirstMbr	指定した識別子を持つ最初のメンバを探す
	SeekNextMbr	指定した識別子を持つ次のメンバを探す
②	EraseMbr	指定したメンバを削除
	BeginNewMbr	メンバの新規出力開始を宣言
	BeginUpdateMbr	メンバの更新出力開始を宣言
	EndMbr	メンバの出力終了を宣言
	WritePnt	内部モデル中の１点をファイルへ出力
	WriteStrk	内部モデル中の１画をファイルへ出力
	WriteWrt	内部モデル中のすべての画をファイルへ出力
	WritePos	xy座標値をファイルへ出力
	WriteEOStrk	画の終わりを示す制御点を出力
③	WriteEOWrt	ｽﾄﾛｰｸﾃﾞｰﾀの終わりを示す制御点を出力
	WriteInvalidPnt	出点(ﾀﾌﾞｯﾄの読みとり範囲を越えた点)を出力
⑥	ReadPnt	１点を内部モデルへ入力
	ReadHeadStrk	１画を内部モデルの最初の画として入力
	ReadTailStrk	１画を内部モデルの最終画として入力
	ReadPrevStrk	１画を内部モデルの１つ前の画として入力
	ReadNextStrk	１画を内部モデルの次の画として入力
	ReadWrt	全ての画を内部モデルへ入力
	ReadPos	ファイルよりxy座標値を入力
	ReadEOStrk	画の終わりを示す制御点を読み飛ばす
	ReadEOWrt	ｽﾄﾛｰｸﾃﾞｰﾀの終わりを示す制御点を読み飛ばす
	ReadInvalidPnt	出点を読み飛ばす
	EOStrk	ファイル中の画の終わりを検出
	EOWrt	ファイル中のｽﾄﾛｰｸﾃﾞｰﾀの終わりを検出
	IsInvalidPnt	出点を検出
④⑤	WriteStrkAttr	１画の物理（論理）属性をファイルへ出力
⑦⑧	ReadStrkAttr	１画の物理（論理）属性をファイルより入力
⑨	WriteStrkPhyAttr	ストロークの物理属性を出力
	ReadStrkPhyAttr	ストロークの物理属性を入力
⑩	WriteStrkLogAttr	ストロークの論理属性を出力
	ReadStrkLogAttr	ストロークの論理属性を入力
	WriteByte	１バイトの数値をファイルへ出力
	WriteWord	１ワードの数値をファイルへ出力
	WriteLongint	２ワードの数値をファイルへ出力
	WriteStr	文字列をファイルへ出力
⑪	WriteEOStr	改行を出力
	WriteEOText	テキストの終わりを示す制御文字を出力
⑭	ReadByte	１バイトの数値をファイルより入力
	ReadWord	１ワードの数値をファイルより入力
	ReadLongint	２ワードの数値をファイルより入力
	ReadStr	文字列をファイルより入力
	ReadEOStr	改行を読み飛ばす
	ReadEOText	テキストの終わりを示す制御文字を読み飛ばす
	EOStr	改行を検出
	EOText	テキストの終わりを検出
⑫⑬	WriteTextAttr	ﾃｷｽﾄ１行の物理（論理）属性をﾌｧｲﾙへ出力
⑮⑯	ReadTextAttr	ﾃｷｽﾄ１行の物理（論理）属性をﾌｧｲﾙより入力
⑰	WriteTextPhyAttr	テキストの物理属性を出力
	ReadTextPhyAttr	テキストの論理属性を入力
⑱	WriteTextLogAttr	テキストの論理属性を出力
	ReadTextLogAttr	テキストの論理属性を入力