デザイン文字の自動生成法（その2）
――そのアルゴリズムとツール開発への研究――

長　江　貞　彦

3. 明朝体

明朝体は日本では最もよく使用され、最も読みやすい書体とされている。明朝体の描画には、漢字スケルトン・ピーススケルトン合成・法線ベクトル法を取り入れたスケルトン・エレメント合成法を用いる。

3.1　明朝体のエレメント

漢字スケルトンを重ねた明朝体の主なエレメントを図8に示す。角ゴシック体に比べて複雑な形状を有するエレメントが多い。“横線”が“縦線”に

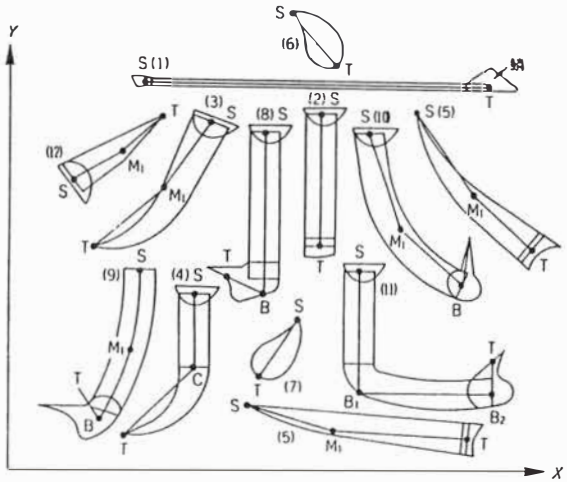


図8　明朝体のエレメント（写真9を参照）

比べて細く、終端部に鱗とよばれる三角形に似た形状のセリフが付加される。多くのエレメントは開始部に右側に突き出したセリフを有する。同図のエレメントの番号は、漢字スケルトン・データ作成上の規定とエレメントの判別の項でのエレメントの番号と一致している。また、スケルトンの記号も各エレメントの説明での記号と一致している。

3.2　システム構成

漢字スケルトン・ピーススケルトン合成・法線ベクトル法を取り入れたスケルトン・エレメント合成法による明朝体描画システムは、以下の４部より構成される。構成を図９に示す。

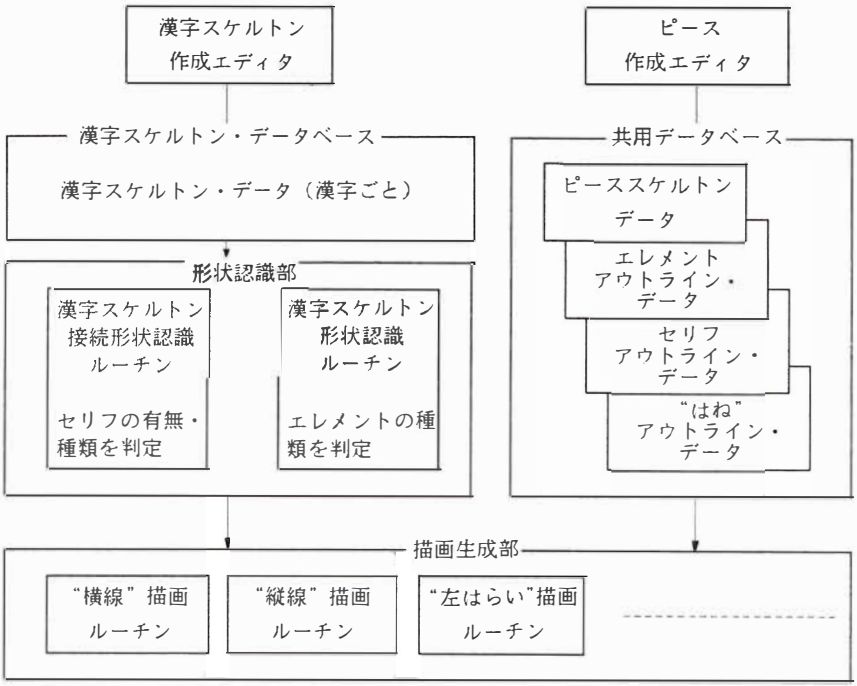


図９　文字発生用システムの構成

(1)　漢字スケルトン・データベース

漢字スケルトン・データベースは漢字ごとのスケルトン・データで構成されている。この漢字ごとのスケルトンを今後、漢字スケルトンとよぶことにする。各漢字スケルトン・データは、文字の左下を原点にとり、漢字スケルトン作成エディタの100×100の$X-Y$座標上でストロークの開始点、屈曲点、カーブ中間点、終了点の座標値とペンアップ情報を筆順で作成する。本システムではペンアップ情報として（−2）を挿入している。（−2）は通常座標値として取り得ないので、ペンアップ情報として用いることができる。

(2)　共用データベース

共用データベースは、セリフなどのピースのアウトライン・データや、“はらい”の主部などのピーススケルトン・データで構成される。これらはすべての漢字に共用され、必要なときに随時用いられる。

各ピース・アウトライン・データは輪郭線をサンプリングした座標点列で構成されている（後述する)。描画生成部で再現するときは2次B−Spline曲線を用いている。描画の際、例えば“はね”の先端点のように1つのスプライン曲線では連続的に描画できない点は、それを示す情報をデータに挿入する。本システムでは（−888, −888）を挿入している。ここに（−888, −888）は通常座標値としては現れないので用いることができる。

(3)　形状認識部

漢字スケルトンの形状を認識する部分で、以下の2つよりなる。

1）　スケルトン形状認識部（エレメント判定部）

明朝体では、各ストロークのスケルトンの形状や方向とエレメントの種類の間に密接な関係がある。そこで、筆順で配列に格納された漢字スケルトン・データを順々に座標値の増減を調べることにより、漢字スケルトンの形状や方向を自動認識し、エレメントの種類を判定する。その自動認識を容易にするため、漢字スケルトンを作成する段階で各エレメントについて後に詳述するような規定を設けている。その規定はごく自然なものであり、漢字スケ

ルトン作成に支障をきたすものではない。

2)　スケルトン接続形状認識部（セリフ判定部）

明朝体では、ストロークどうしの接続形状とセリフの種類に法則がみられる。そこで、**写真２**に示すようスケルトンの接続形状を自動認識し、セリフの種類を判定する。

具体的にはアルゴリズム上では１つの漢字の漢字スケルトンの中で、すべての水平線の両端点の座標値を水平線専用の配列に、すべての垂直線の両端点の座標値を垂直線専用の配列に格納する。この配列は漢字スケルトン・データの配列と同じ配列番号で参照できるようにしておき、何もデータが存在しないところにはそれを示す０を代入しておく。こうしておいてから水平線と垂直線の接続関係を総当たりで調べる。そして明朝体の規則に従ってセリフの有無、存在するならばその種類を決定し、その情報をセリフの有無や種類の情報を格納するための１次元配列に代入する。この配列も漢字スケルトン・データの配列と同じ配列番号で参照できるようにしておき、セリフが存在しないところにはそれを示す０を代入しておく。このように配列番号をそろえておくことでデータの参照を容易にしている。

(4)　描画生成部

形状認識部からのエレメントやセリフの種類の情報や漢字スケルトン・データの位置情報を用いて、各エレメントを各エレメント描画ルーチンで描画生成する。前述した漢字の階層構造を用いて、各エレメント描画ルーチンは１つ以上のピース描画ルーチンで構成されている。ただし、異なったエレメント描画ルーチンが同一のピース描画ルーチンを共用している場合もある。描画のとき必要に応じて共用データベースのデータを用いる。

3.3　漢字スケルトン・データ作成上の規定とエレメントの判別

漢字スケルトン・データは［0～99、0～99］の x–y 座標上で、各ストロークの特徴点（開始点、屈曲点、カーブ中間点、終了点）の座標値やペン

アップ情報を筆順通りにとる。各エレメントについては以下の規定に従う。規定を設ける理由は、システムが各エレメントのデータ点の座標値の変化、およびデータ点の数を情報としてエレメントの種類を判別するためである。（以下の各エレメントについての説明は図9の記号を用いて行う。）

(1)　横線

横線は書始めがやや太く、主部は細く一定の太さで、終端部に鱗とよばれる三角形に似た形状の図形が付加される。鱗もセリフの一種である。したがって、コンピュータで描画する場合3つのピースに分割し、開始部のセリフ、主部、鱗を合成して表示する。漢字スケルトン作成時には、主部の開始点S (Starting Point) と終了点T (Terminal Point) のみをプロットする。コンピュータで横線を自動検出するときは水平になっていること、すなわちy 座標値が一定であることを用いる。これを数式で表せば、$S(x_s,y_s)$、$T(x_t,y_t)$として、

$$y_s = y_t$$

となる。

(2)　縦線

縦線は書始めの部分はセリフの一種である右側への突出部を有し、主部は一定の太さで描き、終端部はやや特殊な形状を有する（これも広義のセリフである)。したがって、コンピュータで描画する場合、3つのピースに分割し、セリフ、主部、終端部を合成して表示する。漢字スケルトン作成時には、主部の開始点Sと終了点Tのみをプロットする。コンピュータで縦線を自動検出するときは垂直になっていること、すなわちx座標値が一定であることを用いる。これを数式を用いて表せば、$S(x_s,y_s)$、$T(x_t,y_t)$として、

$$x_s = x_t$$

となる。

(3)　左はらい

“左はらい”は、書始めの部分にセリフをもっており、そのセリフと主部

の２つのピースに分割して描画する。漢字スケルトン作成時には、主部の開始点$S(=M_0)$と終了点$T(=M_n)$以外にその中間部に１つ以上のカーブ中間点$M_i(i=1, 2, \cdots, n-1)$をとり、計３点以上とする。主部の描画には、漢字スケルトン・ピーススケルトン合成・法線ベクトル法を用いる。

コンピュータで“左はらい”を自動検出する場合は、M_iを原点とみなしたとき、M_{i+1}が第３象限に存在すること（ただし、x軸、y軸は含まない）を用いる。この条件を数式を用いて表せば、$S(x_s,y_s)$、$T(x_t,y_t)$、$M_i(x_i,y_i)$として以下のようになる。

$$x_{i+1}-x_i<0,\ y_{i+1}-y_i<0$$
$$(i=0, 1, \cdots, n-1)$$

あるいは、

$$x_s=x_0>\cdots>x_{i-1}>x_i>x_{i+1}>\cdots>x_n=x_t$$
$$y_s=y_0>\cdots>y_{i-1}>y_i>y_{i+1}>\cdots>y_n=y_t$$

(4)　縦はらい

“縦はらい”は、縦線に“左はらい”が継続する。書始めの部分にセリフを保持し、そのセリフと縦線部および、はらい部の３つのピースに分割して描画する。漢字スケルトン作成時には、縦線部の開始点S、縦線部とはらい部の接続点$C(=M_0)$、終了点$T(=M_n)$の合計３点とし、それ以外に、点Cと点Tの中間部に１つ以上カーブ中間点M_i（$i=1, 2, \cdots, n-1$）をとってもよく、その場合は計３点以上となる。はらい部の描画には漢字スケルトン・ピーススケルトン合成・法線ベクトル法を用いる。

コンピュータで“縦はらい”を自動検出するときは、①線分SCが垂直になっていること、②Mを原点とみなしたとき、Mが第３象限に存在すること（ただしx軸、y軸は含まない）を用いる。この条件を数式に用いて表せば、$S(x_s,y_s)$、$C(x_c,y_c)$、$M_i(x_i,y_i)$、$T(x_i,,y)$として以下のようになる。

①　$x_s=x_c$

② $x_{i+1}-x_i<0, y_{i+1}-y_i<0$

$(i=0, 1, \cdots, n-1)$

あるいは、

$x_c=x_0>\cdots>x_{i-1}>x_i>x_{i+1}>\cdots>x_n=x_t$

$y_c=y_0>\cdots>y_{i-1}>y_i>y_{i+1}>\cdots>y_n=y_t$

(5) 右はらい

“右はらい”は、開始部は細く終端部に近づくに従って太くなり、終端部はやや特殊な形をしている。したがって、主部と終端部のセリフの2つのピースに分割して描画する。漢字スケルトン作成時には、主部の開始点$S(=M_0)$、終了点$T(=M_n)$以外にその中間部に1つ以上カーブ中間点M_i $(i=1, 2, \cdots, n-1)$をとり、計3点以上とする。主部の描画には、漢字スケルトン・ピーススケルトン合成・法線ベクトル法を用いる。

コンピュータで“右はらい”を自動検出する場合は、M_iを原点とみなしたとき、M_{i+1}が第4象限に存在すること（ただしx軸、y軸は含まない）を用いる。この条件を数式に用いて表せば、$S(x_s, y_s)$、$T(x_t, y_t)$、M_i (x_i, y_i)として以下のようになる。

$x_{i+1}-x_i>0, y_{i+1}-y_i<0$

$(i=0, 1, \cdots, n-1)$

あるいは、

$x_s=x_0<\cdots<x_{i+1}<x_i<x_{i+1}<\cdots<x_n=x_t$

$y_s=y_0>\cdots>y_{i-1}>y_i>y_{i+1}>\cdots>y_n=y_t$

(6) 点

“点”は開始部が細く、終端部が丸みを帯びている。漢字スケルトン・データ作成時には開始点Sと終了点Tの座標のみをとる。コンピュータに“点”を自動検出させる場合、開始点Sを原点とみなしたとき、終了点Tが第4象限にあること（ただしx軸、y軸は含まない）を用いる。この条件を数式に用いて表せば、$S(x_s, y_s)$、$T(x_t, y_t)$、として以下のようになる。

$x_t - x_s > 0,\ y_t - y_s < 0$

(7)　逆点

“逆点”の形状は“点”と相似形であるが、表示角度が異なる。漢字スケルトン・データ作成時には開始点Sと終了点Tの座標のみをとる。またコンピュータに“逆点”を自動検出させる場合、開始点Sを原点とみなしたとき、終了点Tが第３象限にあること（ただしx軸、y軸は含まない）を用いる。この条件を数式で表せば、$S(x_s,y_s)$、$T(x_t,y_t)$、として以下のようになる。

$x_t - x_s < 0,\ y_t - y_s < 0$

(8)　縦線はね

“縦線はね”は“縦線”から左にはねるものである。開始部にはセリフをもっている。コンピュータで描画する場合、３つのピースに分割し、セリフ、縦線部、はね部を合成表示する。漢字スケルトン作成時には、開始点S、屈曲点B(Bending Point)、終了点Tの３点をとる。

コンピュータに“縦線はね”を自動認識させる場合、①縦線部が垂直であること、②屈曲点を原点とみなしたとき終了点が第2象限に存在すること(x軸を含む）を用いる。これを数式を用いて表すと、$S(x_s,y_s)$、$T(x_t,y_t)$、$B(x_b,y_b)$として以下のようになる。

①　$x_s = x_b$

②　$x_t - x_b < 0,\ y_t - y_b \geqq 0$

(9)　左そりはね

“左そりはね”は、それだけで１つのストロークを形成することはなく、“横線”に続いて描画される。したがって、“左そりはね”の開始点のセリフは同時に“横線”の終了点のセリフである。システムでは便宜上これを“横線”に属するものとしており、“左そりはね”のみの漢字スケルトン入力からはセリフは描画されない。したがって、コンピュータで描画する場合２つのピースから構成され、主部、はね部を合成表示する。漢字スケルトン

作成時には、開始点S、屈曲点B、終了点Tの3点の他に開始点Sと屈曲点Bの間に1点以上の中間点M_i（$i=1, 2, \cdots, n-1$）をとり、計4点以上とする。主部の描画には漢字スケルトン・ピーススケルトン合成・法線ベクトル法を用いる。

コンピュータで“左そりはね”を自動検出する場合、①M_iを原点とみなしたとき、M_{i+1}が第3象限に存在すること（x軸、y軸は含まない）、②屈曲点を原点とみなしたとき終了点が第2象限に存在すること（x軸を含む）を用いる。この条件を数式を用いて表せば、$S(x_s, y_s)$、$T(x_t, y_t)$、$B(x_b, y_b)$、$M_i(x_i, y_i)$として以下のようになる。

① $x_{i+1}-x_i<0, y_{i+1}-y_i<0$

$(i=0, 1, \cdots, n-1)$

あるいは、

$x_s=x_0>\cdots>x_{i-1}>x_b$

$y_s=y_0>\cdots>y_{i-1}>y_i>y_{i+1}>\cdots>y_n=y_b$

② $x_t-x_b<0, y_t-y_b\geqq 0$

(10) 右そりはね

“右そりはね”は開始点にセリフをもっている。そこで、コンピュータで描画する場合、3つのピースに分割し、セリフ、主部、はね部を合成表示する。漢字スケルトン作成時には、開始点S、屈曲点B、終了点Tの3点の他に開始点$S(=M_0)$と屈曲点$B(=M_n)$の間に1点以上の中間点$M_i(i=1, 2, \cdots, n-1)$をとり、計4点以上とする。主部の描画には漢字スケルトン・ピーススケルトン合成・法線ベクトル法を用いる。

コンピュータで“右そりはね”を自動検出する場合、①M_iを原点とみなしたときM_{i+1}が第4象限に存在すること（ただしx軸、y軸は含まない）、②屈曲点Bを原点とみなしたとき終了点Tが第1，2象限に存在すること（ただしx軸を含まない）を用いる。この条件を数式を用いて表せば、$S(x_s, y_s)$、$T(x_t, y_t)$、$B(x_b, y_b)$、$M_i(x_i, y_i)$として以下のようになる。

①　$x_{i+1}-x_i>0,\ y_{i+1}-y_i<0$

$(i=0,\ 1,\ \cdots,n-1)$

あるいは、

$x_s=x_0<\cdots<x_{i-1}<x_i<x_{i+1}<\cdots<x_n=x_b$

$y_s=y_0>\cdots>y_{i-1}>y_i>y_{i+1}>\cdots>y_n=y_b$

②　$y_t-y_b>0$

⑾　曲げはね

“曲げはね”は開始点にセリフをもっている。コンピュータで描画する場合４つのピースに分割し、セリフ、縦線部、曲げ部、ハネ部を合成表示する。漢字スケルトン作成時には、開始点S、第１屈曲点B_1、第２屈曲点B_2、終了点Tの４点をとる。曲げ部の描画には漢字スケルトン・ピーススケルトン合成・法線ベクトル法を用いる。

コンピュータで“曲げはね”を自動検出するときは、①線分SB_1が垂直になっていること、すなわち開始点Sと第１屈曲点B_1のx座標値が等しいこと、②線分B_1B_2が水平になっていること、すなわちB_1とB_2のｙ座標が等しいこと、③第2屈曲点B_2を原点とみなしたとき終点Tが第１，２象限に存在すること(x軸を含まない）を用いる。この条件を数式を用いて表せば、$S(x_s,y_s)$、$T(x_t,y_t)$、$B_1(x_{b1},y_{b1})$、$B_2(x_{b2},y_{b2})$として以下のようになる。

①　$x_s=x_{b1}$

②　$y_{b1}=y_{b2}$

③　$y_t-y_b>0$

⑿　はね上げ

“はね上げ”は書始めの部分にセリフをもっており、そのセリフと主部の２つのピースに分割して描画する。漢字スケルトン作成時には、主部の開始点$S(=M_0)$と終了点$T(=M_n)$以外にその中間部に１つ以上の点$M_i(i=1,\ 2,\ \cdots,n-1)$を取り、計３点以上とする。主部の描画には漢字スケルトン・

ピーススケルトン合成・法線ベクトル法を用いる。

コンピュータで“はな上げ”を自動検出する場合、M_iを原点とみなしたとき、M_{i+1}が第1象限に存在すること（ただしx軸を含み、y軸は含まない）を用いる。この条件を数式を用いて表せば、$S(x_s,y_s)$、$T(x_t,y_t)$、$M_i(x_i,y_i)$として以下のようになる。

$$x_{i+1}-x_i>0, y_{i+1}-y_i\geqq 0$$
$$(i=0,\ 1,\ \cdots, n-1)$$

あるいは、

$$x_s=x_0<\cdots<x_{i-1}<x_i<x_{i+1}<\cdots<x_n=x_t$$
$$y_s=y_0\leqq\cdots\leqq y_{i-1}\leqq y_i\leqq y_{i+1}\leqq\cdots\leqq y_n=y_t$$

3. 4　ピースの形状と描画方法

ピースをその形状と描画方法で分類すると、大きく次の3つに分類できる。

①長方形のピース。漢字スケルトン・データの位置情報のみを用いて描画する。（例:“縦線”主部、“横線”主部、“縦線はね”縦線部、“縦はらい”縦線部、“曲げはね”縦線部）

②複雑な輪郭線をもつピース。漢字スケルトン・データの位置情報を用い、エレメントまたはセリフのアウトライン・データを共用データベースから呼び出して、拡大・縮小・回転して合成し、2次B-Spline曲線で補間しつつ描画するピース。（例:“点”、“逆点”、“鱗”、セリフの“はね”）

③中心スケルトンを決定でき、そのスケルトンがカーブしており、長さや傾斜角度にさまざまなバリエーションが存在するピース。漢字スケルトン・データの位置情報を用い、漢字スケルトン・ピーススケルトン合成・法線ベクトル法で描画する。（例:“左はらい”主部、“右はらい”主部、“縦はらい”はらい部、“左そりはね”主部、“右そりはね”主部、“はね上げ”主部、“曲げはね”曲げ部）

これらのうち、①については説明するまでもないので②③について後に詳

述する。漢字スケルトンと明朝体描画例を**写真3**に、そのピースを上記の①②③に分類したものを**図10**に示す。また、セリフの形状にはさまざまなものがある。例えば**写真４**のように好みの形状のセリフのアウトライン・データをピース作成エディタで作成し、データファイルを交換することにより、セリフの形状を変えることができる。

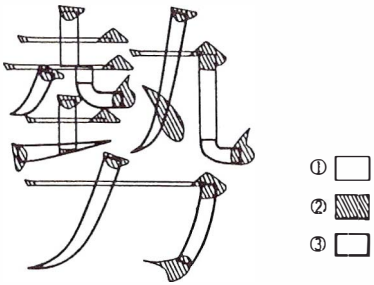


図10　ピースを①②●に分類した明朝体例

3.5　漢字スケルトン・ピーススケルトン合成・法線ベクトル法による描画過程について

左右はらいの主部、左右そりはねの主部、“縦はらい”のはらい部、“はね上げ”の主部、“曲げはね”の曲げ部は、中心曲線が明確に決定でき、長さや傾斜角度にさまざまなものが存在する。このようなピースの場合、中心曲線のサンプル点列をピーススケルトン・データとして開始点を原点にして、共用データベースに作成しておく。そして、該当するピースを描画する場合に、必要なピーススケルトン・データを共用データベースから取り出す。

描画過程は次の通りである。すなわち、**図11**の①のように、漢字スケルトンの開始点Sと終了点Tのx軸方向の距離D_xとy軸方向の距離D_yを計算する。また、同図の②のようにピース・スケルトンの開始点と終了点のx軸方向の距離L_xとy軸方向の距離L_yも計算する。さらに同図③のようにx軸方向、y軸方向のピーススケルトンに対する漢字スケルトンの距離の比$D_x/L_x/D_y/L_y$を計算し、その比をピーススケルトン・データに掛けることにより、x軸方向、y軸方向それぞれ独立に拡大・縮小を行い、ピーススケルト

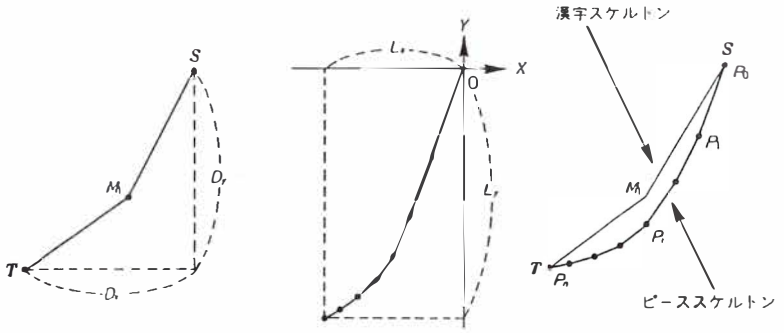


図11　ピーススケルトンを漢字スケルトンへ合成する過程（左はらいの場合）

ンの始終点が漢字スケルトンの始終点に一致するように合成する。

次に、ピース・スケルトンのデータ点列$\boldsymbol{P}_i\,(i=0,1,\cdots,n)$における法線ベクトル$\boldsymbol{N}_i\,(i=0,1,\cdots,n)$を求める。ここで、法線ベクトルは角ゴシック体の平行線描画法のところで述べた手法で求める。法線ベクトルはピーススケルトンをはさんで両側にとるが、大きさを等しくとるのでまとめて$\boldsymbol{N}_i$で表す。各法線ベクトルの大きさ$|\boldsymbol{N}_i|$はピースの種類によって決定の仕方が異なる。“左そりはね”の主部、“右そりはね”の主部、“曲げはね”の曲げ部についてはすべてのデータ点上で同一の大きさをとる。この状態を図12に示す。

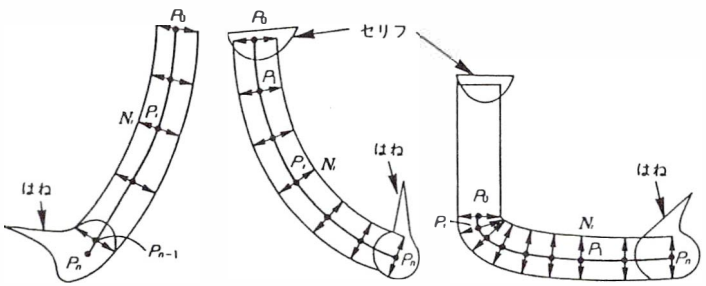


図12　法線ベクトルの大きさが全データ点上で等しいピースを含むエレメント

“左はらい”、“はね上げ”の主部、および“縦はらい”のはらい部については、開始点でピースの幅が太く終了点に近づくほど幅が狭くなり、終点で０となるので次式で定義する。

$$|\boldsymbol{N}_i| = \frac{(n-1)}{n} |\boldsymbol{N}_0| \quad (0 \leqq i \leqq n)$$

また“右はらい”の主部については、開始点でピースの幅が０、終了点に近づくに従って太くなるので次式で定義する。

$$|\boldsymbol{N}_i| = \frac{i}{n} |\boldsymbol{N}_n| \quad (0 \leqq i \leqq n)$$

上の式における $|\boldsymbol{N}_0|$、$|\boldsymbol{N}_n|$ はシステム起動時に入力する明朝体漢字のストロークの太さのパラメータから自動的に計算され適切な値が設定される。また、ピーススケルトンのデータ点の個数 n はいくつであろうとシステムが自動的にカウントし対処する。そして各ベクトルの先端点を求め、２次B-Spline曲線で結ぶことによりピースの輪郭線を生成する。この状態を図13に示す。

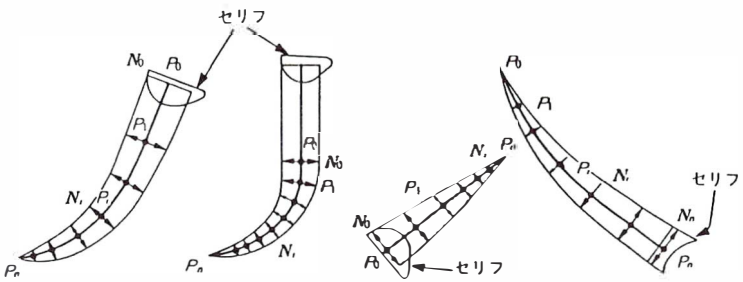


図13　法線ベクトルの大きさがデータ点によって異なるピースを含むエレメント

ここでは以上の手法を「漢字スケルトン・ピーススケルトン合成・法線ベクトル法」と命名する。ピーススケルトンはピース作成エディタ上でマウスを用いて作成する。

3.6　漢字スケルトン・ピーススケルトン合成・法線ベクトル法の特徴

漢字スケルトン・ピーススケルトン合成・法線ベクトル法の長所は以下の通りである。

1)　さまざまな長さ、傾きのピースを同一のピーススケルトン・データにより生成することが可能である。したがって、データ・サイズの縮小につながる。

2)　漢字スケルトンは、ストロークの始終点、屈曲点の位置情報の供給源、およびエレメントの種類判別に用いられるだけなので、判別に必要最小限のデータ点があればよい。したがって、漢字スケルトンの作成効率を向上させることができる。

3)　法線ベクトルにストロークの太さを決定するパラメータを掛けることにより、太さを自由に変化させることができる。

もし漢字スケルトン・ピーススケルトン合成・法線ベクトル法を用いないとすれば、以下の方法が考えられる。

i)　漢字スケルトンのデータ点から直接法線ベクトルを計算してピースの輪郭線を生成する方法

ii)　さまざまな長さ、傾き、曲率をもつピースのアウトライン・データを共用データベースに作成しておき、漢字スケルトン・データからピースの長さ、傾き、曲率を自動的にあるいは漢字スケルトン・データに追加された情報から認識し、最も適したピースのアウトライン・データを選んできて、輪郭線を生成する方法

iii)　ピースのアウトラインを関数式で定義する方法

ここでi)の手法については、漢字ごとに該当するスケルトンにおいてデータ点を多くかつ丁寧にとる必要がある。したがって、漢字スケルトン・データ作成に多くの時間と労力を必要とする。漢字の数が非常に多いことを考えると、その時間と労力は累積され膨大になる。よってはなはだ非効率的な手法である。ただし、1つ1つのピースごとの微妙な曲率の相違を表すこ

とができる。

次にii)の手法は、さまざまな長さ、傾き、曲線をもつピースのアウトライン・データを作成する必要があるため、時間と労力が必要となり共用データベースのサイズも大きくなる。ピースのアウトライン・データの種類数を減らせば、適切と思われる形状に最も近い形状のアウトライン・データを選択して拡大・縮小した後、漢字スケルトンに合成することになるが、x軸方向とy軸方向の拡大率が一般には異なるので、形が歪んでしまう可能性がある。また、個々のピースのアウトライン・データは、ストロークの太さを変化させることができない。したがって、ストロークの太さ別にピースのアウトライン・データを作成する必要がある。

さらにiii)の手法は、ピースごとのデータを作成する代わりに数式で形状を決定してしまう方法である。したがって、ピースの種類によって関数式が異なるので、最も適切な関数式を見いだすのに多くの検討を要する。また、ピースによっては1つの関数式では表せないアウトラインをもつものもあるので、手軽に用いられない手法である。ただし“はらい”については、筆者は１つの関数式で表現できると考えており、今後の研究課題である。

このような点から、漢字スケルトン・ピーススケルトン合成・法線ベクトル法は比較的有効な方法であると考える。ただし、現在のところ、漢字スケルトンからはストロークの曲率を制御できないという短所がある。これは、漢字スケルトンはストロークの始終点の位置情報のみを用いており、例えば“左はらい”では中間点の座標値は全く反映されていない。中間点はそれをプロットすることによりデータ点数が合計３点以上になり、エレメントの種類判別時にデータ点数が２点である。“逆点”と区別できるようにするためである。この短所を解決するには、中間点を通過するようなピーススケルトンを自動的に生成するようなアルゴリズムの開発が必要であり、今後の課題である。

3.7　縦線部に接続し漢字スケルトン・ピーススケルトン合成・法線ベクトルル法で描画するピースのスケルトン

“縦はらい”のはらい部および“曲げはね”の曲げ部はいずれも縦線部に接続している（例えば図12の“曲げはね”、図13の“縦はらい”を参照)。

このようなピースのピーススケルトン・データを作成するとき、開始点P_0とその次の点P_1を両端とする線分P_0P_1がy軸に平行になるようにP_0、P_1をとれば、開始点P_0における法線ベクトルがx軸に平行となるので、縦線部に滑らかに接続できる。さもなければ、縦線部との接続点で間隔が生ずる。

3.8　漢字スケルトン・ピーススケルトン合成・法線ベクトル法で描画するピースに付属するセリフの描画

“縦線”や“横線”のセリフは共用データベースのアウトライン・データを拡大・縮小して描画すればよく、回転の1次変換を施す必要はないが、漢字スケルトン・ピーススケルトン合成・法線ベクトル法で描かれたピースに付属するセリフについては、そのつど回転角度の正弦と余弦を計算し、回転の1次変換を施してから描画する必要がある。この対象となるのは“左はらい”“はね上げ”“右はらい”“右そりはね”のセリフである（図12の“右そりはね”、図13の“左はらい”“はね上げ”“右はらい”)。

“左はらい”“はね上げ”“右そりはね”のストローク開始点のセリフについては、ピーススケルトンを漢字スケルトンに合成した後、ピーススケルトンの第1点（開始点）P_0と第2点P_1を結ぶことによってできる線分P_0P_1の回転角の正弦と余弦を計算し回転の1次変換を行う。

“右はらい”のストローク終了点のセリフのデータについては、ピーススケルトンを漢字スケルトンに合成した後、ピーススケルトンを漢字スケルトンに合成した後、ピーススケルトンの第n点（最終点）P_nと第$[n-1]$点（最終点の1つ手前の点）P_{n-1}を結ぶことによってできる線分P_nP_{n-1}の回

転角の正弦と余弦を計算し、回転の１次変換を行う。

3.9　漢字スケルトン・ピーススケルトン合成・法線ベクトル法で描画するピースに付属する“はね”の描画

漢字スケルトン・ピーススケルトン合成・法線ベクトル法で描かれたピースに付属する“はね”は、セリフと異なり回転の１次変換を行う必要はないが、どのようなスローブを有するピースにも滑らかに接続するように“はね”のアウトライン・データは原点付近で円形の形状をもたせている。“曲げはな”“右そりはね”“左そりはね”の“はね”はその例である（図12)。

また“左そりはね”の“はね”は、原点が“はね”の重心よりかなりずれた右上方に位置しているため、漢字スケルトンの主部の終点に“はね”の原点を合成して描画すると“はね”の全体が意図したより下方に位置してしまい不自然である。特に“力”という漢字の場合、“左はらい”と“左そりはね”の下端の位置をほぼ等しくする必要がある（厳密には“左そりはね”の下端の位置は“左はらい”の下端の位置よりやや下げた方が安定して見える）。この解決策として、主部の描画の際に漢字スケルトンにピーススケルトンを合成した後、主部はピーススケルトンの第n点（最終点）P_nまで描かず第［$n-1$］点（最終点の1つ手前の点）P_{n-1}まで描き、その点に“はね”の原点を位置させて“はね”を描画する（同じく図12を参照)。したがって主部のピーススケルトンを作成するとき、このことに留意しながら第n点P_nと第［$n-1$］点P_{n-1}の相対位置を決定する必要がある。

3.10　ストロークの太さの変化に伴うピース間の接続点の移動

ストロークの太さを変化させると、ピース間の接続点も変化させなければならない場合が存在する。それは図14に示すように“縦線はね”の主部と“はね”の接続点、および“曲げはね”の縦線部と曲げ部の接続点である。

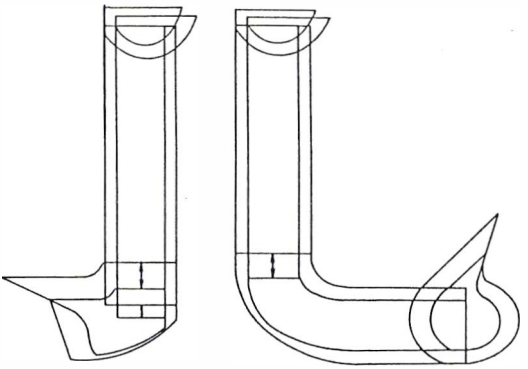


図14　ストロークの太さの変化に伴うピース間の接続点の移動

(1)　“縦線はね”

一般にストロークの太さを変化させるとき、“はね”のアウトライン・データにはx方向、y方向の両方向に拡大・縮小のパラメータを掛ける。この場合、“はね”のアウトライン・データの原点の位置によっては、拡大・縮小したときに“はね”の下端の位置が大きく変化してしまう。“縦線はね”の場合、例えば“月”や“周”という漢字を例にとると、“左はらい”と“縦線はね”の下端の位置がそろっていなければ不自然である。そこで、“はね”のアウトライン・データ作成時に原点を下端付近にとり、拡大・縮小すなわちストロークの太さを変えても下端の位置はほとんど変化しないようにしている。また、このとき“はね”を拡大するに従って“はね”の上端の位置は上方に移動するので、それに伴い縦線部の終点の位置を上方に移動させなくてはならない。そのために、縦線部の終点の位置をストロークの太さの関数とし、ストロークを太くするに従って縦線部の下端（終了点）の位置が上方に移動するようにアルゴリズムを組んである。

(2)　“曲げはね”

“曲げはね”の縦線部と曲げ部の接続点は、ストロークの太さの増加に

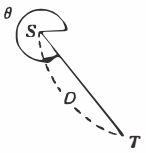
従って上方に移動させなければならない。さもなければストロークを太くしていくに従い、接続点付近の曲げ部の曲率半径が短くなり過ぎて、見かけ上、急カーブとなり不自然である。そこで、接続点の位置をストロークの太さの関数とし、ストロークを太くするに従って接続点の位置が上方に移動するようにアルゴリズムを組んである。

3.11　長さ、太さ、角度が可変で複雑な形状を有するエレメントの描画

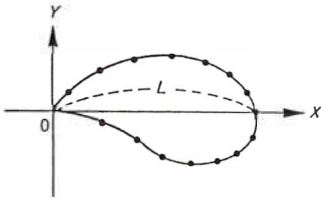
“点”“逆点”はセリフや“はね”と同様に複雑なアウトラインを有するので、共用データベースのアウトライン・データを用いて描画する。そして、“点”“逆点”は形状が同じであるから、アウトライン・データは同一のデータを用いている。ただしセリフや“はね”はx、y両方向に等倍率で拡大・縮小を施せばよいのに対し、“点”“逆転”は長軸方向の長さ、角度の違いによりさまざまなものが存在し、またストロークの太さ、すなわち長軸と直角方向の幅を変化できるようにしておく必要がある。そこで“点”“逆点”のアウトライン・データは、開始点を原点とした座標で、長軸をx軸に一致させたかたちで共用データベースに保存されている。

このアウトライン・データを漢字スケルトンに合成する過程は次の通りである。

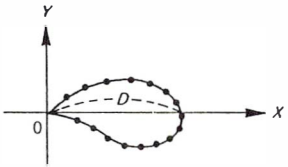
まず、図15の①に示すように漢字スケルトン・データの“点”“逆点”の開始点Sと終了点Tの距離Dを計算する。また同図②のようにアウトライン・データの長軸の長さLは既知であるから、比率D/Lを計算しアウトライン・データの各点のｘ座標値に掛け、長さ変換を施す。さらに同図③のようにｙ座標値にはストロークの太さを決定するパラメータを掛け、太さ変換を施す。次に、漢字スケルトン・データの“点”“逆点”の開始点Sと終了点Tを通る直線STについて、x軸に平行でSを通る直線を基準線とする回転角θ（図15の①参照）の正弦と余弦を計算し、回転の1次変換を施す。これにSの座標値を加えてB-Spline曲線で結べば、同図④のように開始点S



① 漢字スケルトン



② アウトライン・データ



③ 長さ・太さ変換後

④ 合成後

図15 “点”“逆点”のアウトライン・データを漢字スケルトンに合成する過程

と終了点Tに一致した“点”“逆点”が描画できる。

3.12 考察と吟味

漢字スケルトン・ピーススケルトン合成・法線ベクトル法を取り入れたスケルトン・エレメント合成法の特徴をまとめると、以下のようになる。

(1) いま**写真5**に示すように任意の大きさ（ただし極小サイズは除く）の漢字を描画生成でき、x軸方向、y軸方向のサイズを独立に変化させることができるので、**写真6**の状態で縦横比を変えることができる。この場合、漢字スケルトンをx軸方向、y軸方向に拡大・縮小してからエレメントを合成するので、マッピングによって縦横比を変える従来の方法では、x軸またはy軸に平行でない輪郭線をもつピース（例えば“はらい”）の形状が歪んだり、太さが変化してしまう問題点を完全に解決した。

(2) **写真7**のようにストロークの太さを変化させることが可能である。

(3) セリフなど一部の形状をアウトライン・データを取り替えることによ

り、容易に変換できる（**写真４**を参照）。また、新しいアウトライン・データを作成することにより、新しい書体を創造できる。

(4)　共用データベースにより可能なかぎりのデータの共有化を図り、漢字ごとのデータはスケルトンで構成されているので大幅なデータ・サイズの縮小化を実現している。

(5)　漢字スケルトン・データはストロークの始終点、屈曲点の座標値とペンアップ情報だけで構成されており、それ以外の情報は全く付加されていないので非常に汎用性が高い。したがって、他の書体に対して適切なエレメント描画ルーチンを作成することにより、同一のスケルトン・データを用いてさまざまな書体を描画できるシステムに発展させることができる。また、そうすることにより、書体が異なるごとに漢字全体のアウトライン・データを保持する必要がないので、アウトライン方式に比べてデータ・サイズの縮小につながる。

(6)　漢字を筆順に描画していくのでこれは**写真８**からも判る通り描画過程自体が筆順教育用のCAIアニメーションとなっている。漢字をエレメントやピースに分割して描画しているので、一部のエレメント、ピースの長さや形状を変化させることが可能であり、さまざまな文字アニメーションへの応用が考えられる。

以上のような特徴があげられるが、描画された文字の品質からいうとアウトライン法と比べやや劣る。これは次のような原因になる。

i)　ピースを合成して描画しているため、ピースとピースの接続点で、アウトラインが滑らかになるべきところで滑らかにならない箇所が生ずる。

ii)　アウトライン・フォントでは、１つの文字の中でストロークが特に込み入っている箇所では文字の黒みと空白の面積のバランスを保つため、ストロークの太さを細めにしたり、鱗などのセリフの大きさをわずかに小さくするなどの対策を施している。

ここでi)の解決法としては、ピースとピースのアウトラインの交点を産出

し、スプライン曲線などを用いてスムージングを施すようなアルゴリズムの開発が必要である。

次にii)の解決法としては、漢字スケルトン・データからストロークの間隔や密度を算出し、ストロークの太さやセリフの大きさを自動調節するようなアルゴリズムの開発が必要である。

この他、描画速度もアウトライン法に劣る。このようにいくつかの欠点は存在するが、先にあげた1)〜6)の特徴はアウトライン法では得難いものである。特に本手法はデザイナーが新書体を創造する際に大きな威力を発揮すると考えている。

以上、明朝体の文字に対してもゴジック体（前回の「デザイン文字の自動生成法（その1）」を参照）と同様、本研究のアルゴリズムが正しいことを**写真9**に示す。これと**図8**との比較により完成度の高さが実証されたといえる。

4．装飾文字

ゴシック体、明朝体、毛筆体などのコンピュータによる描画は最近、産官学ともに盛んになってきているが、装飾文字の描画はまだほとんど研究されていない。ここでは日本の伝統的な装飾文字の一つである髭文字を対象とし、髭の描画法について述べる。髭文字とは、籠字、相撲字などの肉太の書体に、髭とよばれるさまざまな装飾を施したものである。髭の種類には“木口そぎ”“なが毛”“いな穂”“大いな穂”“七五三”“あら毛”“火焔”“吹き流し”“つの”などがある。各髭の発生アルゴリズムは、次のことに留意しながら作成されている。

①可能なかぎり少ないデータ点数から効率的に髭を発生させる。

②パラメータの数値を変えることにより、同一データからさまざまなバリエーションの髭が得られるようにする。

③マン・マシン・インタフェースを考慮し、データ点の座標はできるかぎり

マウスを用いて入力できるようにする。

以下に各髭のコンピュータによる発生法について述べる。ただし、装飾を施す前の文字はアウトライン法で描画している。

4.1　“木口そぎ”　“なが毛”

文字の“はらい”“はね”あるいは書始めの筆圧が低い部分などに、数本の細い線の切込みでそぐことを“木口そぎ”という。“木口そぎ”は一般に曲線でそぐが、その描画を容易にしかつデータの圧縮を行うため２つの曲線データを与え、その間に線形に内挿曲線を発生させる内挿曲線発生アルゴリズムを開発した。すなわち、図16のように$m+1$本の曲線群の第０番目の曲線上のサンプル点列A_{0l}、…、A_{0j}、…、A_{0n}および第m番目の曲線上のサンプル点列A_{ml}、…、A_{mj}、…、A_{mn}をデータとするとき、第i番目の曲線上のサンプル点列の第j番目の点A_{ij}は、

$$A_{ij}=\frac{iA_{mj}+(m-i)A_{0j}}{m}$$

$(0\leqq i\leqq \mathrm{m},1\leqq \mathrm{j}\leqq n)$

で与えられる。サンプル点列を３次補間曲線で補間しながら内挿曲線を描く。内挿曲線の太さ、本数は自由に決定できる。写真10に付加前図を、写真11に付加後の字体例を示す。

また、細長く文字を構成する場合、長い髭を施すのを“なが毛”という。“なが毛”の作成は、“木口そぎ”の方法を用いる。毛筆の“かすれ”のような効果を表

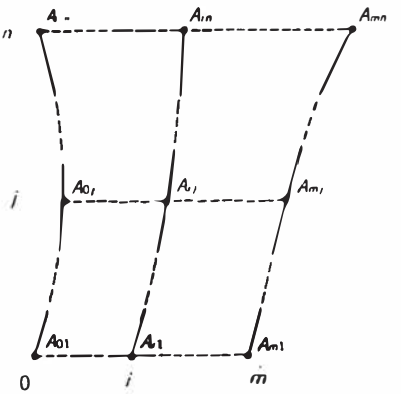


図16　内挿曲線の発生アルゴリズム

現できる。図17に付加後の例を示す。

図17　“かすれ”

4.2　“いな穂”“大いな穂”“七五三”“あら毛”

文字の輪郭の外側に数本の髭を施すものを“いな穂”という。“いな穂”はその曲線群の最も外側の2曲線のサンプル点をデータとし、まず内挿曲線発生アルゴリズムによって内挿曲線を描く。

単に髭を付加しただけでは、髭と文字の輪郭線との接続部は不自然である。そこで、図18のように隣り合う髭のそれぞれの中心曲線のサンプル点の第1点どうし、および第2点どうしをそれぞれ結んだ補助線$A_{i1}A_{i+11}$、$A_{i2}A_{i+12}$ $(0 \leq i \leq m-1)$を考える。これらの補助線と髭の輪郭線との交点R_{i1}、L_{i+11}、R_{i2}、L_{i+12} $(0 \leq i \leq m-1)$の順にL_{i+12}の順に2次B−Splineの曲線で結ぶ。2次B−Splineの曲線は、それらの中点を通過する。さらに、このB−Splineの曲線を移動させることにより、接続部の無色領域を塗りつぶす。以上のアルゴリズムをいな穂発生アルゴリズムと命名する。“いな穂”の長いものを“大いな穂”というここに写真12に“いな穂”“大いな穂”を併用した付加後の字体例を示す。

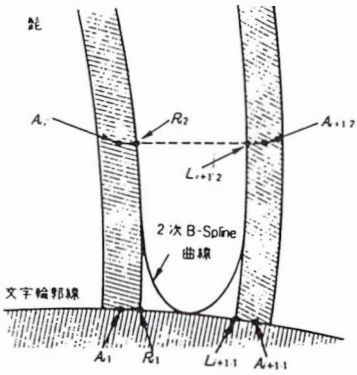


図18　“いな穂の生成”

「7、5、3」の数は、昔からめでたい数として伝えられてきた。そこで、文字の3箇所に“いな穂”をそれぞれ7、5、3本ずつ施したものを、特に

“七五三”という。“七五三”は、日本では慶事を表す文字によく用いられる。**写真13**に付加後の例を示す。

髭を数本の太い線にすると硬い味をもった文字になる。この先の角張った太い髭を“あら毛”という。“あら毛”の作成は内挿曲線発生アルゴリズムを用い、いな穂発生アルゴリズムは用いない。ここに**写真14**に処理後の例を示す。

4.3 “火焔”“吹き流し”“つの”

“炎”を文字に付加することにより、文字の意味を強調したり視覚的な暗示を与えることができる。そのような髭文字を“火焔文字”もしくは単に“火焔”とよんでいる。**図19**に“炎”の作成図を示す。中心曲線のサンプル点列$\boldsymbol{P}_i\,(i=0,\ 1,\ \cdots, n)$をデータとし、各点における法線ベクトル$\boldsymbol{N}_i\,(i=0,\ 1,\ \cdots, n)$を求める。法線ベクトルは中心曲線をはさんで両側にとるが、大きさを等しくとるので、まとめて$\boldsymbol{N}_i$で表す。その大きさ$|\boldsymbol{N}_i|$は“炎”の根元で太く、先端にいくほど細くなるので次式で算出する。

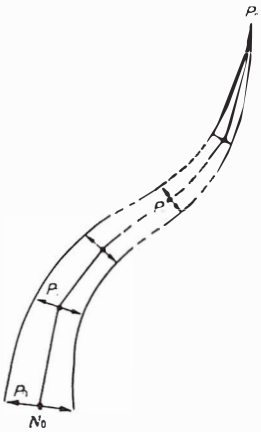


図19　“炎”生成図

$$|\boldsymbol{N}_i| = \frac{(n-i)}{n}\ |\boldsymbol{N}_0| \quad (0 \leqq i \leqq n)$$

ここで、$|\boldsymbol{N}_0|$はユーザーが入力する“炎”の太さのパラメータで決定される。また、データ点の個数nはいくつであろうともシステムが自動的にカウントして対処する。これより各ベクトルの先端点を求め、２次 B-Spline 曲線で結ぶ。

実際の作業では、もとの文字を画面上に表示し、ユーザーもしくはデザイ

ナーがマウスを用いて好みの場所にデータ点をプロットするので、さまざまな形の“炎”を個人の感性を生かしてデザインすることが可能である。“炎”の太さは“炎”ごとに数値で入力するので、そのつど太さを変えることも可能である。付加前と付加後の例を**写真15**および**写真16**に示す。

“炎”のデータ点列を直線にすれば“つの”になる（**写真17**）。“つの”を同一方向に延ばせば、“吹き流し”とよばれる、文字が風に吹かれているような漢字を表現できる（**写真18**）。

4.4 考　察

髭文字の研究はこれまでのところ、対象を髭の創成アルゴリズムに限定しており、もとになる籠字などの文字については研究の対象外としてきたが、今後、エレメントに分割することによりスケルトンから自動生成することを検討したい。

また“炎”は、フラクタルを用いて“炎”の途中から小さな“炎”が枝分かれするさまの自動的な生成を試みたい。

5．今後への発展

最近 DTP の研究は民間企業で大変盛んであり、その成果が発表されつつある。民間企業の場合、文字フォントは大部分がアウトライン・フォントを用いている。これは文字の輪郭線の品質が良いためである。

本研究の漢字スケルトン・ピーススケルトン合成・法線ベクトル法を取り入れたスケルトン・エレメント合成法による漢字フォントは、民間のアウトライン・フォントと比べると残念ながら品質が劣る。これはピースに分割して描画しているため、ピースどうしの接続が完全には滑らかにはならないこと、可能なかぎりデータの共有化を図っているので、個々の漢字の全体の形状とストロークのバランス性から要求されるエレメントの微妙な輪郭線の違いを表現することができないためである。したがって、本手法はプロ用の電

算写植機などには向かない。しかし、パーソナルユースのDTPソフトウエアなどには、データ・サイズの縮小化が望まれることから、例えば**写真19**の例にあるようにカンバン文字のレイアウトや、**写真20**のようなネオン文字および**写真21**のような照明付（カンバン）文字のシミレーション等に利用価値が高いと考えられる。

また、今後DTPは、明朝体やゴシック体などの既存のフォントを描画生成する画一的な研究から、新しいフォントを作り出したり文字をデザインするという創造的な方向へ移行していくものと考えられる。そのときスケルトン・エレメント合成法は、部分的なアウトライン・データの交換が可能なため新書体の創造という面から有力な手法であり、新たな可能性を生み出すと考えられる。

また、装飾文字の研究は、日本ではまだあまり行われていない。DTPは個人で個性を生かした印刷物の作製ができるようにならなければ、機動性のあるパーソナルコンピュータ、あるいはそれに代わる小型コンピュータを使用している長所が生かせない。DTPには表現力の豊かさが必要であり、そのためには書体の豊富さ、あるいは新しい書体をデザインできることが必要である。その中で個人の感性を文字の中に表現していける装飾文字の開発は、今後ますます重要になってくるものと考える。

なお、本研究は筆者が大阪府立大学総合科学部情報科学科に在職中の折、修士学生の曽我真人君（現在は和歌山大学に勤務）を指導したときのテーマを引き続き発展させたものである。同君はもとより、共同研究で協力してくれた辻合秀一君（現在は近畿大学に勤務）に対し、ここに改めて深謝の意を表したい。また、本研究のアルゴリズムのプログラム化やそのテストランに協力を願った近畿大学文芸学部芸術学科における研修員の栗栖直也氏ならびに資料の整理や写真の撮影にご尽力いただいた近畿大学文芸学部芸術学科・四回生の大場由紀枝と時松良枝の両氏に心からの感謝を表明する。

参考文献

1)　辻合秀一、曽我真人、中谷吉次、長江貞彦：「レタリングCAD（その1）――CAL(Computer Aided Lettering)のアルゴリズム開発――」、日本図学会、図学研究第41号、pp.19−24、1987

2)　曽我真人、長江貞彦：「レタリングCAD――角ゴシック体――」、第3回ソフトウェア・コンファレンス・プロシーディングス、大阪科学技術センター、pp.187−190、1987

3)　曽我真人、辻合秀一、長江貞彦：「レタリングCAD（その2）――角ゴシック体・明朝体・テキスチャーマッピングを施した文字――」、日本図学会、図学研究第42号、pp. 7−14、1987

4)　辻合秀一、曽我真人、長江貞彦：「レタリングCAD」、NICOGRAPH’87、第3回NICOGRAPH論文コンテスト論文集、pp.148−152、日本コンピュータ・グラフィックス協会、1987

5)　曽我真人、長江貞彦：「骨格ストローク・エレメント合成法による漢字明朝体の生成」、昭和62年電気関係学会関西支部連合大会講演論文集、pp. G271、1987. 11

6)　辻合秀一、曽我真人、長江貞彦：「レタリングCAD（その3）――特殊文字について――」、日本図学会、図学研究第43号、1987

7)　辻合秀一、曽我真人、中谷吉次、長江貞彦：レタリングCAD（その1）、日本図学会

8)　辻合秀一：大阪府立大学総合科学部計量科学コースにおけるCGおよびCADに教育について−事例報告（1）レタリングCAD−、日本設計製図学会、第15回CAD教育の進め方フォーラム予稿集、pp. 5−8、1986

9)　辻合秀一、長江貞彦：コンピュータ・グラフィックスにおける感性の挿入、図形処理情報センター、PIXEL、pp.121−124、No.4、1986

10)　張憲栄、真田英彦、手塚慶一：「計算機による様々な書体生成に適合する筆触パターンの提案」、電子情報通信学会論文誌D、pp.885−892、Vol. J69−D、No.6、1986

11）内尾文隆、張憲栄、真田英彦、手塚慶一：「階層分解合成法に用いる筆画パラメータの半自動決定システム」、電子情報通信学会論文誌D、pp.893−903、Vol.J69−D、No.6、1986

12）張憲栄、真田英彦、手塚慶一：「漢字楷書毛筆字体の計算機による生成」、電気通信学会論文誌D、pp.599−606、Vol.J67−D5、1984

13）大町尚友：「レタリングエッセンス」、日本文芸社、1986

14）佐藤義雄：「実習グラフィックス」、pp.143−158、アスキー出版、1986

15）日向数夫：「毘文字」、グラフィック社、pp.238−249、1979

16）日向数夫編：「書体総字典」、グラフィック社、1979

17）伊藤晃：「高品位文字フォント概論」、PIXEL、No.64、pp.78−85、1988.1

18）「デスクトップ・パブリッシング」、別冊科学朝日、1987.10

19）Jiarong Li：“Generation of Some Chiniese Characters with Metafont”, TEX for Scientific Documentation, pp.161−170, 1985

20）長江貞彦：「アーチスト支援ＣＧシステム構築への試行」、近畿大学文芸学部論集「文学・芸術・文化」創刊号、pp.63−88、1990. 3

21）長江貞彦：「芸術系学生における図学教育について」、近畿大学文芸学部論集「文学・芸術・文化」第２巻、第１号、pp. 25−44、1990. 6

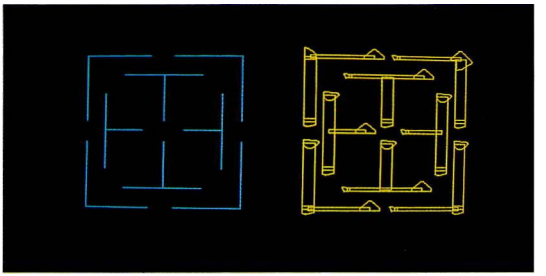


写真2　スケルトンの接続とセリフの有無・種類

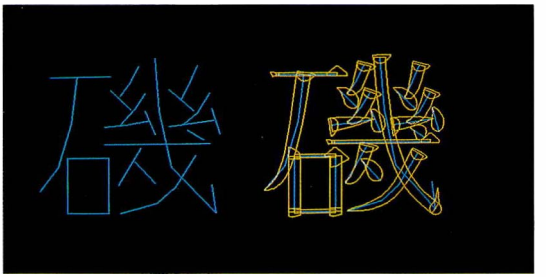


写真3　漢字スケルトンと明朝体描画例

写真4　セリフを変えた明朝体

写真5　明朝体グレーディング

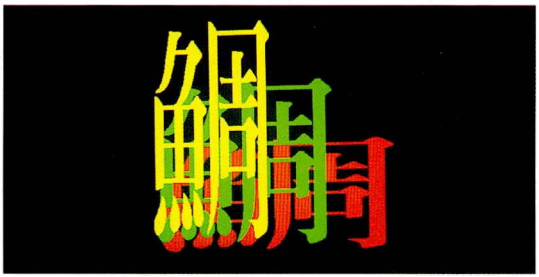


写真６　明朝体の縦横比の変換

写真７　明朝体ストロークの太さ変換

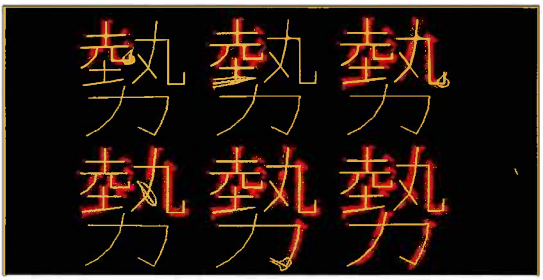


写真８　明朝体の描画課程

写真９　明朝体の自動生成（図８を参照）

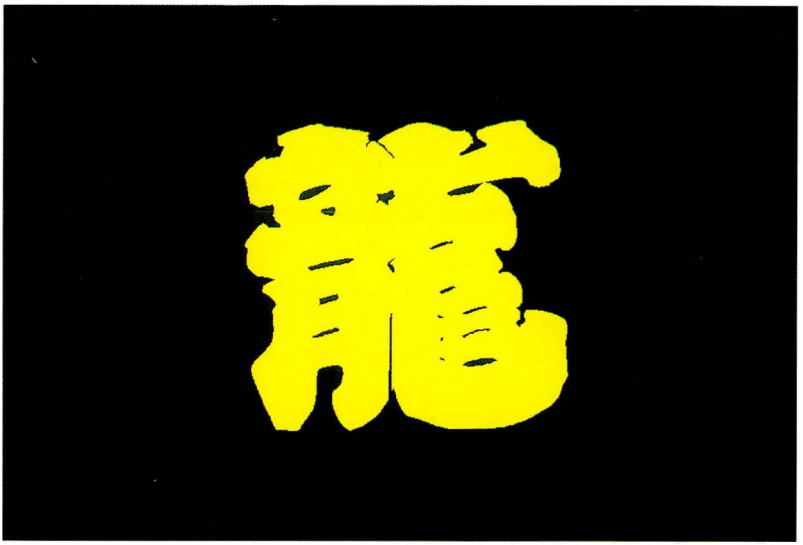


写真10　装飾文字（その1）

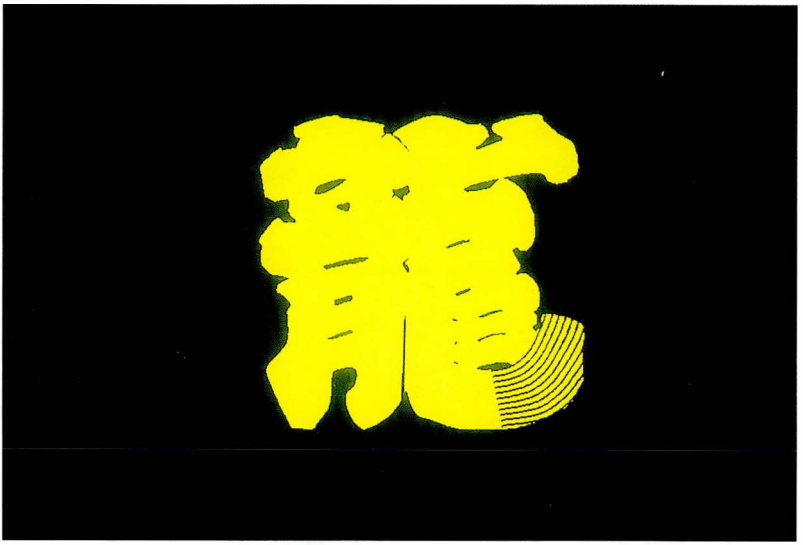


写真11　“木口そぎ”

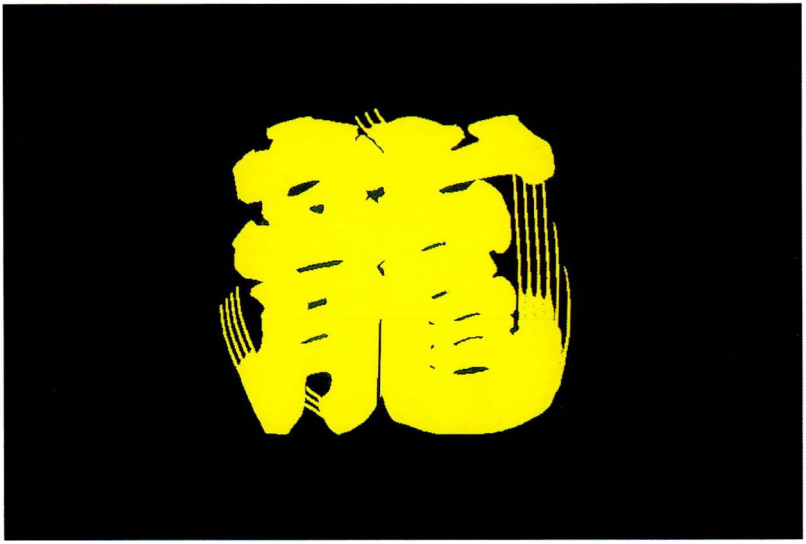


写真12　“いな穂”

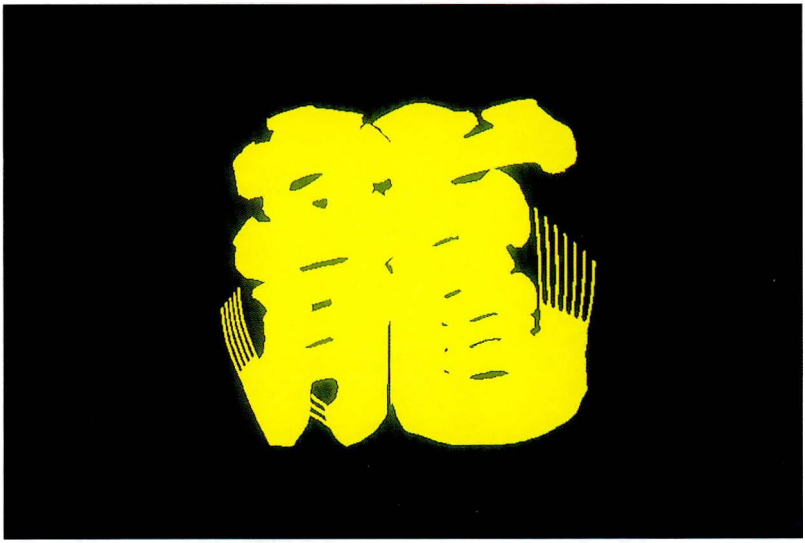


写真13　“七五三”

写真14　“あら毛”

写真15　装飾文字（その2）

写真16　“吹き流し”

写真17　“つの”

写真18　“火焰”

写真19　自然光による反射式（カンバン）文字の例

写真20　自光式（ネオンサイン）文字の例

写真21　透過光式（照明付）文字の例