2次元コンピュータ・グラフィックスのための学習用プログラムの作成

——基本命令から図形描写まで——

広 内 哲 夫

How to Make a Program of 2-Dimentional Computer Graphics for Learnning

——Rang from Programming Basic Routines to Drawing a Picture——

Tetsuo HIROUCHI

A raster scan type of display is suitable for the begginers' learning computer graphics(CG). This type of display generates a many dots on the screen, what are called "picsels". As a result of assembling picsels, a picture or graph is formed on the screen.

But the biginners hardly know the theory of raster scan graphics. It is reported in this paper that the two-dimensional CG program is developed to help the beginners understand easily this theory.

Both using and understanding the program will give them a knowlegde of process of transforming a picture into a set of picsels by function such as difinition of picture, geometry transformation, window-viewport transformation, clipping, generation of line and so on.

1．はじめに

現在，初心者向きのコンピュータ・グラフィックス（CG）といえば，ラスタースキャン型グラフィックスである。このグラフィックスはディスプレイ画面のピクセルと呼ばれる点を1つづつ発光させて，全体で図形・画像を創り出すものである。これは新聞の写真と同じで，近くで見ると点が多数集まって見えるが，遠く離れた所から見ると，1つの図形や絵として見えるという原理を採用している。

簡便なラスタースキャン型ディスプレイでも，一画面に25万6千個ものピクセルが存在する。しかしながら，CGの初心者はこのピクセルの存在に余り気がつかないようである。点（ピクセル）の集まりから1つの線分が描かれ，また線分の集まりから1つの面の色塗りがなされるというラスターグラフィックスの原理を教えると，初めてその存在を理解する。そしてグラフィック・エディター等を使って図形を作成する初心者は，最終的にそのCGソフトウェアによってコントロールされた1つ1つのピクセルの発光から図形が形成されることを知ると，そのソフトウェアの作画アルゴリズムの精巧さに驚くものである。

筆者は大学の文科系の学生のために初等的なラスターグラフィックスを教えている。その経験からすると，一般に学生が興味を示す

3次元の物体表示やその質感を表現するレイトレーシングなどの技術だけを教えるのは，賢明なやり方でないと考える。コンピュータによる作画は，画面という2次元平面が基礎であり，2次元の理論的取り扱いが理解できなくては，CG教育の基礎はぐらついてしまう。

2次元図形はコンピュータでどのように定義されるのか，その図形データはどのようなアルゴリズムによって処理されるのか，そして最終的にどのようにして画面上の1つ1つのピクセルの発光として図形が再現されるのか，等の基本的な原理を学ぶことは，初心者にとって非常に重要なテーマである。

筆者は教育の場でこのようなことを理解させる初心者用の2次元CGプログラムを開発したので報告する。使用言語は，現在のところラスターグラフィックスの学習には最適な日本電気製のN_{88}-BASIC(86)（以降BASICと略記する）を用いる。第2章でBASICのグラフィック機能および一般のCGソフトウェアの持つ必要最小限の機能を示す。第3章ではそれらと同様な機能の基礎作画ルーチンをソフトウェア命令として実現する方法を示す。そして第4章で，ピクセルの発光を統合化された図形描画プログラムによりコントロールすることによって，図形をディスプレイ画面上に描き出す方法を紹介する。

2．コンピュータ・グラフィックスの一般機能

本稿は一般のCGソフトウェアの持つ基礎作画ルーチンをソフトウェア命令として作成することを目的としている。そこで本章では，第3章以降の基本作画ルーチンを作成するうえで必要となる機能の概略を，BASICの機能と対応させて説明する。

2．1 BASICのグラフィック文

BASICに備わっているグラフィック機能の中で，基本的な作画に関するハードウェア命令は，次のステートメントである。

- PSET文……指定の画面位置に存在するピクセルを指定の色で発光させる。
- POINT文……画面のピクセルの位置を指定する（ピクセルは発光させない）。
- LINE文……指定の画面位置から指定の画面位置の間に存在するピクセルを，指定の色で発光させて線分を描く。
- PAINT文……指定の色で発光したピクセルが形成する閉曲線内部の領域のピクセルを，指定の色ですべて発色させて，その領域に色を塗る。

上記の4つのステートメントは，ピクセルという用語を用いてその機能が説明されている。ラスターグラフィックスの〝線は点の集まりから定められる〟という原理により，LINE文の機能は，BASICのハードウェア命令であるPSET文を用いてソフトウェア命令として作り出すことが可能である。また，〝面は直線の集まりによって定められる〟という原理により，PAINT文の機能は，同じくハードウェア命令のLINE文，さらにはPSET文を用いてソフトウェア命令として作り出すことも可能である。

3．1節と3．2節の目的は，LINE文とPAINT文の持つ機能を唯一のハードウェア命令であるPSET文を用いてソフトウェア命令として作り出すことである。

2．2 ウインドウ-ビューポート変換とクリッピング

作画機能以外の基礎機能として，BASICには次の2つのハードウェア命令が存在する。

- WINDOW文……ウインドウを定義する。
- VIEW文……ビューポートを定義

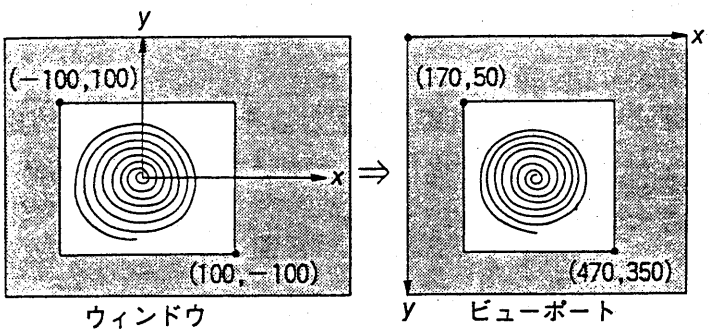


図1　ウィンドウとビューポート

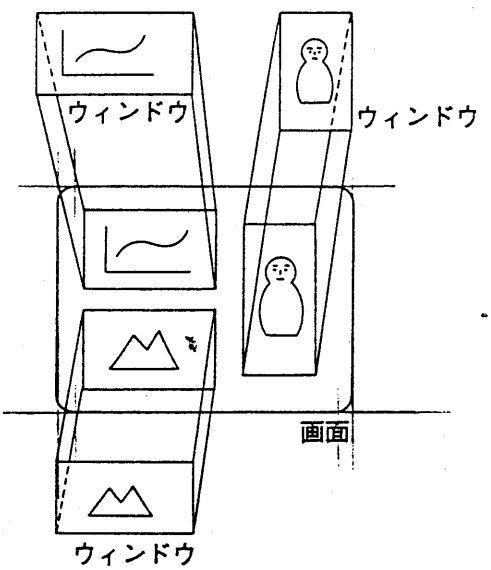


図2　マルチビューポート

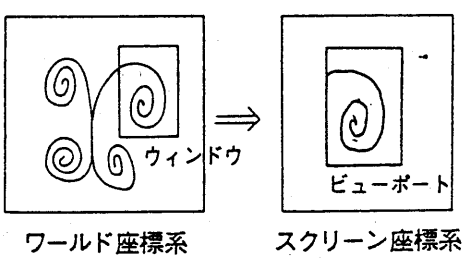


図3　クリッピング

する。

CGにおいては一般に，画面上の座標(ピクセルそのもの) をスクリーン座標と呼び，利用者の定義する座標をワールド座標と呼ぶ。ウインドウとは，広大なワールド座標系の中で画面に写し出したい部分の領域であり，ビューポートとは，ウインドウを実際に写し出す画面上の領域である。ウインドウとビューポートの関係を図1に示す。

WINDOW文とVIEW文がペアで用いられると，それらの文は，ワールド座標系で表現された図形データをスクリーン座標系のデータに自動変換する。これをウインドウ-ビューポート変換と呼ぶ。

またCGでは一般に，図2に示すように，1画面に複数個のウインドウをそれと同数個のビューポートとして設定することができる。これをマルチビューポートと呼ぶ。そのために画面表示の際，ビューポート領域からはみ出した図形を，図3のように刈り取るクリッピングと呼ばれる操作が行われる。VIEW文はこれを行う。

3.3節と3.4節の目的は，WINDOW文とVIEW文と同様な機能の基礎作画ルーチンをソフトウェア命令として作り出すことである。

2.3　幾何変換

CGにおける重要な技術の1つは，図形全体を一定の規則のもとに変形することである。これを幾何変換と呼ぶ。例えば高層ビルを表現する3次元モデルに3次元幾何変換を適用すれば，実際に飛行機に乗って高層ビル群を

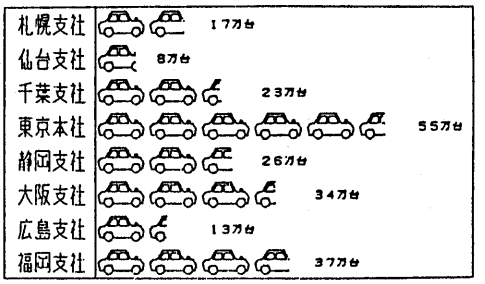


図4　ビジネスグラフの例

斜めに上から，真上から，あるいは遠方上空から眺めたような景観を画面上に作り出すことができる。幾何変換には，平行移動，拡大縮小，反転，せん断，回転等の変換がある。これらの変換は2次元グラフィックといえども，初心者のCG学習には不可欠の課題の1つである。

一般のCGソフトウェアはこの幾何変換のためのハードウェア命令を持っている。残念なことにBASICではこの命令を持っていない。

3．5節の目的はこの幾何変換の基礎作画ルーチンをソフトウェア命令として作成することである。

2．4　図形データの定義

図形を描き出すには，それを構成する図形要素を定義する必要がある。例えば風景を描く場合，その構成要素である家，木，動物，雲などの図形要素が必要である。図4に示すようなビジネス・グラフィックスを作成する場合には，自動車を図形要素として定義しなければならない。

一般のCGソフトウェアにおいては，図形データベースを持っており，このような図形をデータベースに蓄積できるようになっている。しかし，BASICではこのような機能を持っておらず，図形要素の定義および保存は，利用者自身が行わなければならない。

第4章の目的は，論理的にすべての図形の定義ができる一筆書きの図形データの定義方法とその図形描写のプログラムを示すことである。

3．基礎作画ルーチンのアルゴリズム

3．1　直線の作画

Bresenhamは整数の加減算のみを用いて，デジタル直線を描くアルゴリズムを考察した。これはDDA（Digital Differential Analyzer：デジタル微分解析機）と呼ばれる有名な方法である。[2,4,5] 本章ではこの方法を用いてデジタル直線のソフトウェア命令である基礎作画ルーチンを実現する。以下にDDAアルゴリズムを紹介する。

直線の方程式は

$$y=\frac{\Delta y}{\Delta x}x \tag{1}$$

で表わされる。ここで$\Delta y/\Delta x$は直線の傾きである。

あるピクセルの座標を$(x_1,\ y_1)$とし，その$x>0$の方向に隣り合うピクセルの座標を$(x_{i+1},\ y_{i+1})$とすれば，

$$x_{i+1}=x+1 \tag{2}$$

の関係が成り立ち，x_iにおいてyの真の値とy_iの差をe_i（これを誤差項と呼ぶ）とすれば，x_iにおけるy_iの値との差e_{i+1}は，

$$e_{i+1}=e_i+\frac{\Delta x}{\Delta y} \tag{3}$$

で与えられる。そこで，

$$\left.\begin{array}{ll} e_{i+1}\geqq 0.5\text{ならば} & y_{i+1}=y_{i+1} \\ & e_{i+1}=e_{i+1}-1 \\ e_{i+1}<0.5\text{ならば} & y_{i+1}=y_1 \end{array}\right\} \tag{4}$$

とすれば，初期値x_0，y_0，e_0を定めると，式(2)によるx_iの変化に応じて，式(4)からy_iの値を逐次決定することができる。座標$(x_i,\ y_i)$の発光したピクセルの点列が求めるデジタル直線である。これを図5の流れ図に示す。

しかし，式(3)，式(4)には除算および実数演算が含まれているので，この2つの演算を取り除き，整数の加減算のみでデジタル直線を

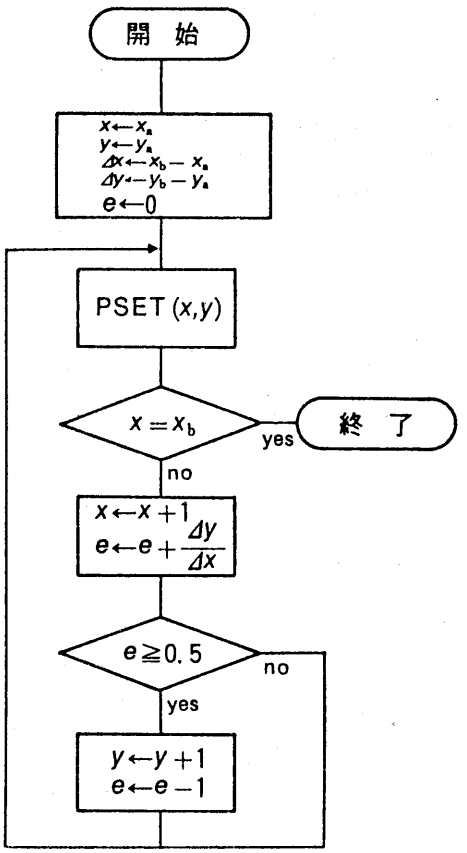


図5　Bresenhamの直線描画アルゴリズム

決定することにする。そこで式(4)の誤差項 e_i の判定を，e_i が0.5より大きいか小さいかで行っていたが，これを新しい誤差項

$$e'_i = e_i - 0.5 \qquad (5)$$

を用いて，e'_i が正か負かを判定することとする。これはさらに新しい誤差項

$$e''_i = 2(e_i - 0.5)\Delta x \qquad (6)$$

を用いても同じことである。

そこで誤差項目 e''_1 を導入し，式(3), 式(4)を次のように変更する。

$$e''_{i+1} = e''_i + 2\Delta y \qquad (7)$$

（ただし $e''_0 = -\Delta x$ ）

$$\left.\begin{array}{ll} e''_{i+1} \geqq 0 \text{ ならば} & y_{i+1} = y_{i+1} \\ & e''_i = e''_i - 2\Delta y \\ e''_{i+1} < 0 \text{ ならば} & y_{i+1} = y_i \end{array}\right\} (8)$$

これが整数化されたDDAアルゴリズムである。

リスト1にこのアルゴリズムを用いて作成された直線描画のソフトウェア命令である基礎作画ルーチンDDAを示す。行番号2100～2170は $|\Delta y/\Delta x| \leqq 1$ の時，行番号2190～2250は $|\Delta y/\Delta x| > 1$ の時，それぞれ利用される。以下にこのサブルーチンの引数およびその内容を示す。

・入力用引数

変数X1, Y1……線分の始点座標

変数X2, Y2……線分の終点座標

・出力用引数

なし

なお，線に色を付けるのは，BASICのステートメントであるPSET文

PSET (X, Y), C

のパラメータCに該当する色コードを指定すればよい。

3．2　色塗り

3．2．1　基本的な考え方

図形に色を塗る考え方の基本は，その内部を色の付いたデジタル直線で隙間なく埋めることである。そうすれば図形内部に存在するピクセルがすべて発光するので，図形に色を塗ったことになる。ラスターグラフィックスの世界では線が点から成り立っているように，面は線から成り立っていると考える。本章では筆者の考案した色塗りのアルゴリズムを紹介し，それを用いて色塗りのソフトウェア命令である基礎作画ルーチンを実現することにする。

色塗りの手続きの概略は次の通りである。図6に示すように，まず直線 l_1 と図形を構成する辺との間で交点 P_1, Q_1 を求める。P_1, Q_1 の間にデジタル直線を描く。そしてこの手順を図6の直線 l_1 から l_n について繰り返し行う。ただし直線 l_1, l_2, ……, l_n から求まるデジタル直線は相接し合うものとする。図形領域のすべてにデジタル直線を描くことが

リスト1

```
2000 '*************************************************************
2010 '*                          DDA サブルーチン                   *
2020 '*************************************************************
2030 *DDA
2040 DEFINT I
2050 IX1=X1:IY1=Y1:IX2=X2:IY2=Y2
2060 IDX=IX2-IX1   :IDY=IY2-IY1
2070 IA=ABS(IDX)   :IB=ABS(IDY)
2080 IX=IX1        :IY=IY1
2090 IF IA>=IB THEN GOTO 2110 ELSE GOTO 2190
2100 '-------------------------------------------------------
2110 IE=-ABS(IDX)
2120 PSET(IX,IY),COL
2130   IF IX=IX2 THEN GOTO 2260
2140   IX=IX+SGN(IDX)
2150   IE=IE+2*IB
2160   IF IE>=0 THEN IY=IY+SGN(IDY):IE=IE-2*IA
2170   GOTO 2120
2180 '-------------------------------------------------------
2190 IE=-ABS(IDY)
2200 PSET(IX,IY),COL
2210   IF IY=IY2 THEN GOTO 2260
2220   IY=IY+SGN(IDY)
2230   IE=IE+2*IA
2240   IF IE>=0 THEN IX=IX+SGN(IDX):IE=IE-2*IB
2250   GOTO 2200
2260 DEFSNG I
2270 RETURN
```

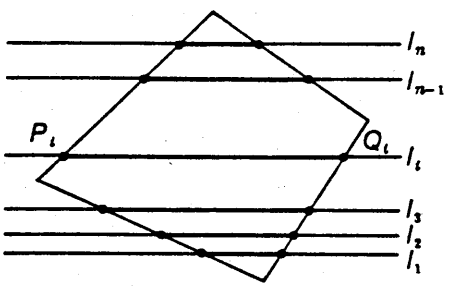


図6　デジタル直線による色塗り

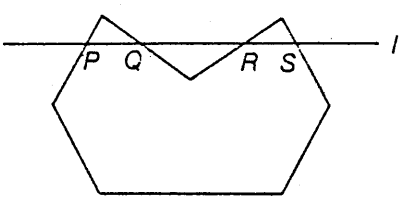


図7　凹多角形の交点

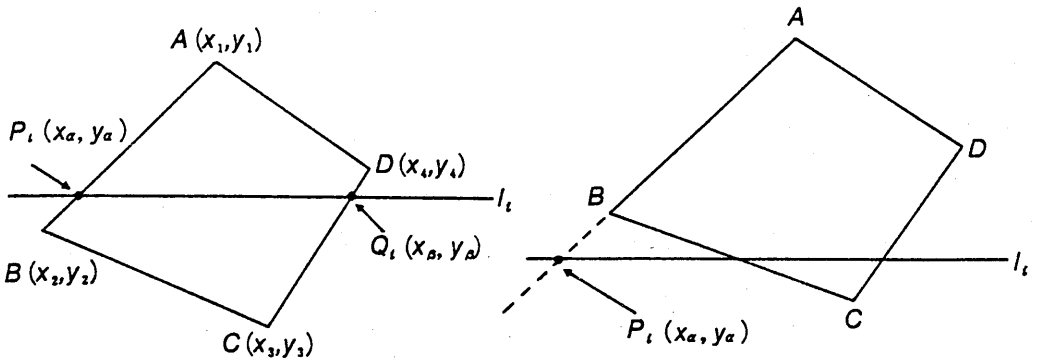
できたら，色が塗れたことになる。

3.2.2　交点の決定

図7に示すように1つの直線lに対する交点が4つ以上求まる図形も存在するが，まず1つの直線lに対して交点が2つだけ求まる簡単な図形を取り上げる。ここでは前者を凹多角形，後者を凸多角形と呼ぶことにする。凹多角形は後で説明するが，それは凸多角形に比べて取り扱いが難しくなる。

図8(1)を用いて，直線l_1と辺との交点を求める方法を説明する。計算が簡単になるように，直線l_1は水平すなわちx軸と平行であるとする。そこで，このl_1をここでは水平直線と呼ぶことにする。水平直線l_1を次のように定める。

$$y=h \tag{9}$$



(1) 条件を満たす点　　(2) 条件を満たさない点

図8　水平直線との交点

図形の頂点A（(x_1, y_1)）と頂点B（x_2, y_2）を結ぶ直線の方程式は

$$y=\frac{y_2-y_1}{x_2-x_1}(x-x_1)+y_1 \qquad (10)$$

と表現される。図8(1)に示すように，線分$\overline{AB}$と水平直線l_iとの交点P_iの座標（x_α, y_α）は，式(9)，式(10)の解から

$$\left.\begin{aligned} x_\alpha&=\frac{x_2-x_1}{y_2-y_1}(h-y_1)+x_1\\ y_\alpha&=h \end{aligned}\right\} \qquad (11)$$

と得られる(ただし$y_1=y_2$の場合には，解は不能または不定となる)。しかし，この座標は，

$$\left.\begin{aligned} x_1\leqq x_\alpha\leqq x_2 \quad &\text{または} \quad x_2\leqq x_\alpha\leqq x_1\\ y_1\leqq y_\alpha\leqq y_2 \quad &\text{または} \quad y_2\leqq y_\alpha\leqq y_1 \end{aligned}\right\} \qquad (12)$$

なる条件を満たす必要がある。というのは，もし満たさない場合には，点P_iは図8(2)に示すような線分$\overline{AB}$の延長線上に存在することになるからである。

同様に頂点Cと頂点Dを結ぶ線分$\overline{CD}$と水平直線l_iとの交点Q_iの座標を次のように求める。

$$\left.\begin{aligned} x_\beta&=\frac{x_4-x_3}{y_4-y_2}(h-y_3)+x_3\\ y_\beta&=h \end{aligned}\right\} \qquad (13)$$

この座標は次の条件を満たす必要がある。

$$\left.\begin{aligned} x_3\leqq x_\beta\leqq x_4 \quad &\text{または} \quad x_4\leqq x_\beta\leqq x_3\\ y_3\leqq y_\beta\leqq y_4 \quad &\text{または} \quad y_4\leqq y_\beta\leqq y_3 \end{aligned}\right\} \qquad (14)$$

ところで，人間は水平直線と交わる辺を一瞬のうちにパターン認識能力を用いて選び出すことができる。それ故，我々は交わらない辺とl_iとの交点を求めることなどは，始めから無意識に除外して，必要な辺とl_iとの交点だけを選択的に求める。この作業は人間が頭の中で行うには容易であるので，選択的に交点を求めるプログラムを作成することは一見簡単そうに思われる。しかし，このようなパターン認識の含まれたプログラムを開発することは非常に大変である。

そこで便法を講じ，水平直線l_iと図形を構成するすべての辺との間で総当たりで交点を求める。そして，この交点の座標の中から条件(12)および(14)を満たす交点のみを選び出す。これが交点P_iとQ_iの座標となる。このようにして求めた点P_iと点Q_iとの間にデジタル直線を描くのである。

3．2．3　問題点

しかし，このような単純なアルゴリズムでは問題が発生する。それは図9に示すような図形では，頂点A，E間にはデジタル直線を描くことができるが，頂点B，D間には描くことができない。これは頂点における交点を

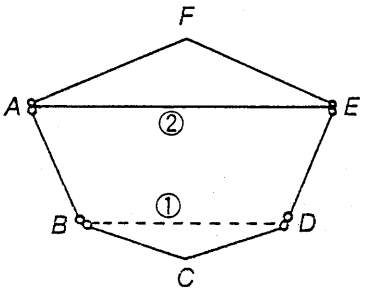


図9　描けないデジタル直線①
と描けるデジタル直線②

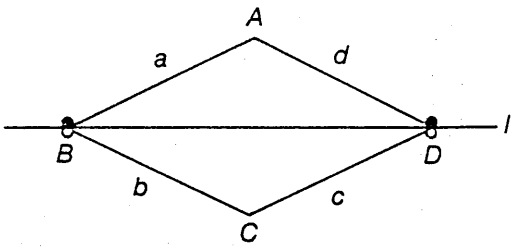


図10　頂点における交点

求める順番に関係する。

例えば図10に示す四角形の場合，頂点B，Dのそれぞれにおいて2つ，合計4つの交点が求まる。その4つの交点とは，頂点B，Cを通る水平線分1とそれぞれの辺a，b，c，dとの交点である。交点を求める順を1とa，1とb，1とc，1とdの組の順とすると，交点のx座標は，頂点B，B，D，Dのx座標であり，それがその順番で求まる。

そこで，デジタル直線は頂点Bから頂点B，また頂点Dから頂点Dへと描かれ，頂点Bから頂点Dへは描かれないことになる。このため図9に示す①のデジタル直線は描くことができない。

次に水平直線1と辺a，b，c，dとの交点をその順番を変えて，1とb，1とc，1とd，1とaの組の順に求める。この場合は，交点のx座標は頂点B，D，D，Bのx座標となり，デジタル直線は頂点Bから頂点Dへ，また頂点Dから頂点Bへと2回描かれる。これが図9の②のデジタル直線を描くことができる理由である。

3．2．4　不要交点の除去

上記に示した問題点を一般化してみよう。頂点において1つの水平直線1に対して得られる交点の数は，図11に示すように3通り存在する。第一は図形の末端の頂点で交点が2つ得られる場合〔Ⅰ〕である。第二はすでに詳しく述べたが，2つの頂点のy座標が同じで交点が4つ得られる場合〔Ⅱ〕である。第三が1つの頂点で交点が2つ，1つの辺で交点が1つ，合計3つ得られる場合〔Ⅲ〕である。

このように水平直線l_1と辺の交点の数が頂点のところで一定しないのは，デジタル直線を描くうえで困ったことである。場合〔Ⅰ〕の時には，1つの頂点に2つの交点があってもよいが，しかし，場合〔Ⅱ〕では，それは都合が悪い。

頂点における不要な交点を除去する規則は次のように定められる。図12に示すように，頂点P_0のy座標をβ_0，また頂点P_0を端点とする2つの辺の他の端点P_1，P_2のy座標をβ_1，β_2とすれば，図12(1)のように水平直線1の点P_0におけるy座標β_0が，次の条件式

$$(\beta_1<\beta_0<\beta_2)\ \text{or}\ (\beta_2<\beta_0<\beta_1) \qquad (15)$$

を成立させるのであれば，後から求めた交点は除去するようにする。また，図12(2)のように，同じく水平直線1の点P_0におけるy座標β_0が，辺の条件式

$$((\beta_1<\beta_0)\ \text{and}\ (\beta_2<\beta_0))$$
$$\text{or}\ ((\beta_1>\beta_0)\ \text{and}\ (\beta_2>\beta_0)) \qquad (16)$$

を成立させるのであれば，後から求めた交点は除去しないようにする。

このアルゴリズムを追加することにより，図9の①のデジタル直線は描くことができるようになる。

3．2．5　凹多角形の場合

図13に示すような凹多角形では，水平直線1と辺との交点は，頂点以外のところでも4

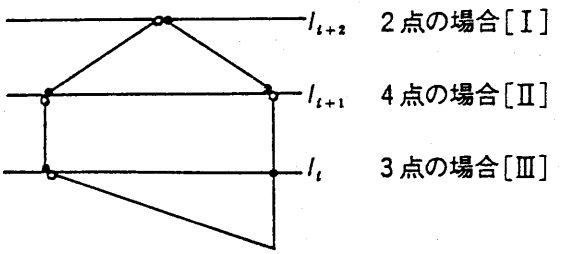
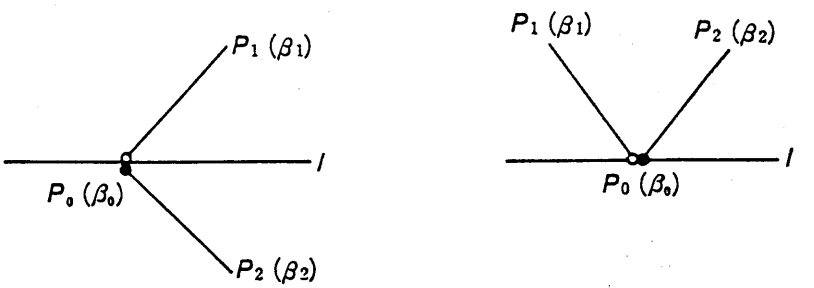
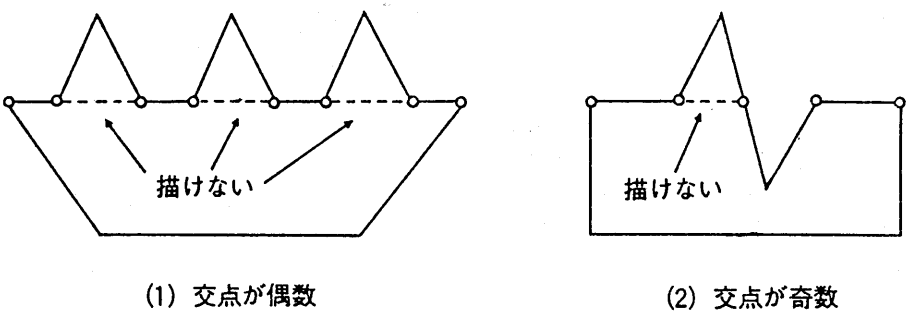


図11　頂点における交点の数の一般化



(1) 交点を除去する場合　　(2) 交点を除去しない場合

図12　交点除去の規則



(1) 交点が偶数　　(2) 交点が奇数

図13　平行線が存在する例

つ以上，しかも必ず偶数個得られる。図13(1)のように交点の得られる順番が，そのx座標の大小の順と同じであるならば，1つの水平直線1から定められるデジタル直線は，交点1と交点2，交点3と交点4，……，交点7と交点8との間で分割されて描かれる。

しかし，図13(2)に示すように，交点は辺を定める頂点の番号の振り方（図中，頂点番号は丸で囲まれた数）によって，その得られる順番が変化する。図(2)の場合，交点3，交点2，交点1，……，交点4のような順で得られるので，それらの交点をその順に2つづつ取り出してデジタル直線を描くことはできない。したがって，一旦，得られた交点のx座標の値を大小順に分類する必要がある。交点がそのx座標の順で分類されれば，交点の座標を大小順に2個づつ取り出すことができるので，個々のデジタル直線を凹図形に対しても描く

ことができる。

3．2．6　平行な辺が存在する時

これまでのアルゴリズムの中で，不都合なことが1つある。それは図14に示すような非常に特殊な図形において，水平直線1と平行な辺が存在する時である。平行な2つの直線の間では，その解は不能あるいは不定なので，交点を求めることができない。この場合は当然デジタル直線を描くことができない。

この問題に対する最も簡単な解決策は，水平直線1のかわり，y軸に平行な垂直直線1′を設定し，x軸上を移動させるアルゴリズムを追加することである。このアルゴリズムはこれまでに述べたものと基本的には全く同じアルゴリズムで，単にx座標とy座標を機械的に入れ替えたものである。効率上〝2度塗り〟の無駄が生じるが，図形の塗られていない部分は確実に塗ることができる。

＊　　＊　　＊　　＊

リスト2に色塗りのソフトウェア命令である基礎作画ルーチンXPAINTを示す。(なおリスト2には水平直線1と平行な辺が存在する場合に対応するルーチンは含まれていない)。行番号3000～3290は，水平直線の交点を求め，デジタル直線を描く部分(Ⅰ)，行番号3300～3430は不要交点を除去する部分(Ⅱ)，行番号3450～3550は交点のx座標を分類する部分（Ⅲ）である。図15に部分（Ⅰ）のルーチンの流れ図を示す。

なお，サブルーチンXPAINTの引数およびその内容を以下に示す。

・入力用引数

配列PX,PY…図形の頂点のx，y座標（頂点は反時計回りの順にセットする）

変数N……頂点の数

変数X,Y……図形内の任意の1点の座標（最初デジタル直線を描き始める点）

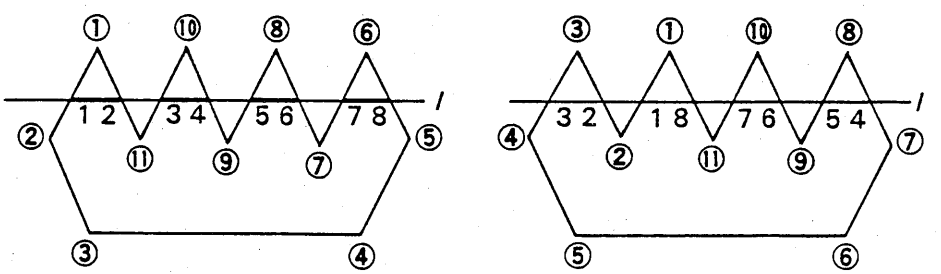
変数GA……インターフェース・ルーチンを作成する時に用いられる引数であり，ここではGAの内容は0として使用

・出力用引数

なし

3．3　ウィンドウ-ビューポート変換

ワールド座標（W座標）からスクリーン座標（S座標）に座標変換する公式は次のようにして求められる。図16に示すように左上隅，右下隅が（a，b），（c，d）の右手系のW座標系におけるW（x，y）が，同じく左上隅，右下隅が（p，q），（r，s）の左手系のS座標系におけるS（x'，y'）に座標変換される場合，W座標系とS座標系におけるそれぞれの線分の比は変換前後において，x軸方向，y軸方向とも常に不変である。そこで



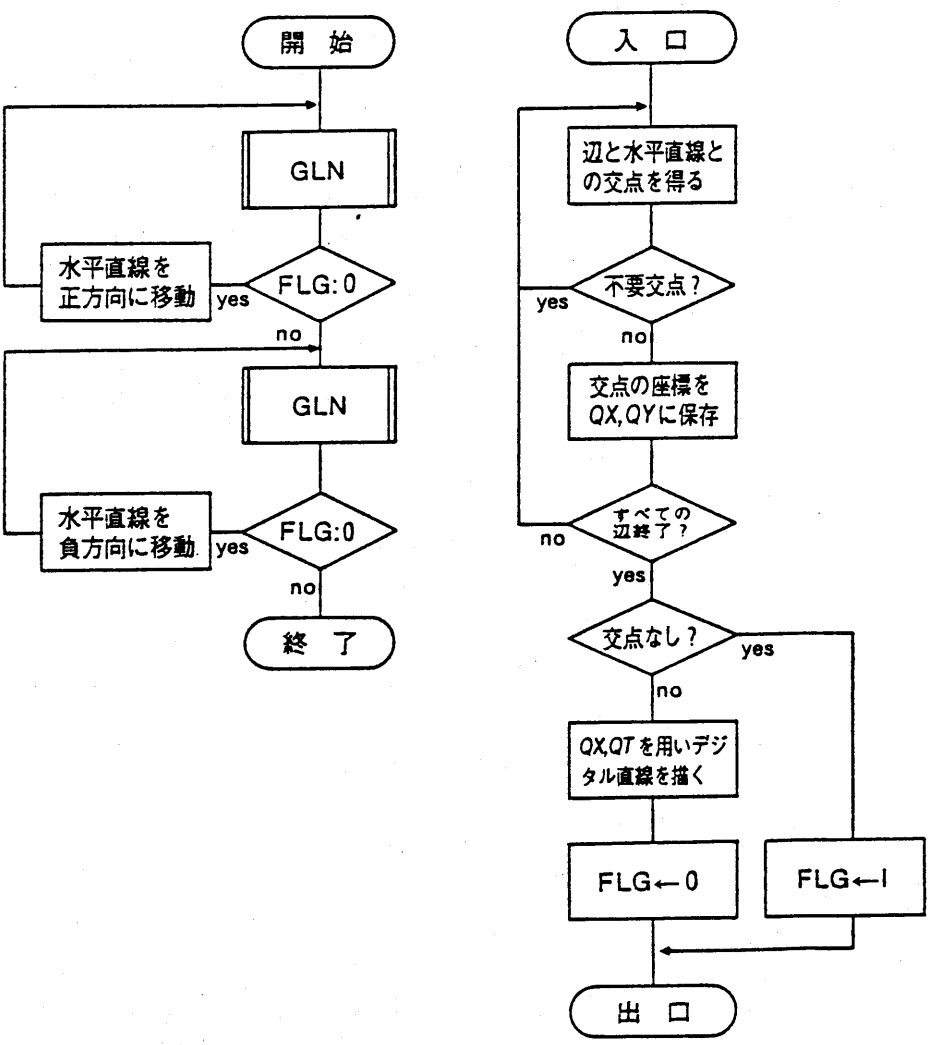
(1)　正常な順番　　(2)　都合の悪い順番

図14　凹多角形における交点の得られる順

リスト2

```
3000 '*******************************************************
3010 '*                      XPAINT ｻﾌﾞﾙｰﾁﾝ                    *
3020 '*******************************************************
3030 *XPAINT
3040 IF PSW=0 THEN DIM QX(20),QY(20):PSW=1
3050 CY=Y
3060 H=CY
3070 GOSUB *GLN
3080   IF FLG=0 THEN H=H+1+GA:GOTO 3070
3090 H=CY
3100 GOSUB *GLN
3110   IF FLG=0 THEN H=H-1-GA:GOTO 3100
3120 RETURN
3130 '-------------------------------------------------------
3140 *GLN
3150 M=0
3160 FOR I=1 TO N
3170   X1=PX(I):Y1=PY(I)
3180   IF I<>N THEN X2=PX(I+1):Y2=PY(I+1) ELSE X2=PX(1):
                                              Y2=PY(1)
3190   IF Y1<>Y2 THEN X=(H-Y1)*(X2-X1)/(Y2-Y1)+X1:Y=H
                 ELSE GOTO 3220
3200   IF X=>X1 AND X=<X2 OR X=>X2 AND X=<X1 THEN
                                              ELSE GOTO 3220
3210   IF Y=>Y1 AND Y=<Y2 OR Y=>Y2 AND Y=<Y1 THEN GOSUB *GCHK
3220 NEXT I
3230 IF M=0 THEN FLG=1:GOTO 3290
3240 GOSUB *GSRT
3250 FOR I=1 TO M STEP 2
3260   X=QX(I  ):Y=QY(I  ):GOSUB *UPOINT:
       X=QX(I+1):Y=QY(I+1):GOSUB *ULINE
3270 NEXT I
3280 FLG=0
3290 RETURN
3300 '-------------------------------------------------------
3310 *GCHK
3320 IF I=N THEN GOTO 3340
3330 P=M:Q=I-1:R=I+1:GOSUB *GSET:GOTO 3380
3340 IF QX(M)=X THEN GOTO 3360
3350 IF QX(1)=X THEN GOTO 3370
3360 P=M:Q=N-1:R=1  :GOSUB *GSET:GOTO 3380
3370 P=1:Q=N  :R=2  :GOSUB *GSET
3380 RETURN
3390 '-------------------------------------------------------
3400 *GSET
3410 IF M=0 THEN M=M+1:QX(M)=X:QY(M)=Y:GOTO 3430
3420 IF NOT (QX(P)=X AND ((Y>PY(Q) AND Y<PY(R)) OR (Y<PY(Q)
                 AND Y>PY(R)))) THEN M=M+1:QX(M)=X:QY(M)=Y
3430 RETURN
3440 '-------------------------------------------------------
3450 *GSRT
3460 FOR J=1 TO M
```

```
3470    VL=QX(J)
3480    ID=J
3490    FOR K=J TO M
3500      IF QX(K)<VL THEN VL=QX(K):ID=K
3510    NEXT K
3520    SWAP QX(J),VL
3530    QX(ID)=VL
3540  NEXT J
3550  RETURN
```



(1) メインプログラム　　(2) GLNサブルーチン

図15　色塗りプログラムの流れ図

x軸，y軸について次の2つの比例式が成立する。

$$\left.\begin{aligned}(a-x):(p-x')&=(x-c):(x'-r)\\(b-y):(q-y')&=(y-d):(y'-s)\end{aligned}\right\}\quad(17)$$

式(17)を整理するとW（x, y）からS（x', yy'）への変換公式は

$$\left.\begin{aligned}x'&=\frac{p-r}{a-c}x+\frac{ar-pc}{a-c}\\y'&=\frac{q-s}{b-d}y+\frac{bs-qd}{b-d}\end{aligned}\right\}\quad(18)$$

となる。

式(18)をBASICの義関数数を用いて表現すれば，

```
DEF    FNX (X)=((SX1-SX2) / (WX1-WX2)* X+(WX1 * SX2-SX1 * WX2) / WX1-WX2)
DEF    FNY (Y)=((SY1-SY2) / (WY1-WY2)* Y+(WY1 * SY2-SY1 * WY2) / WY1-WY2)
```

である。ここで（WX1, WY1），（WX2, WY2），（SX1, SY1），（SX2, SY2），（X, Y），（FNX (X), FNY (Y)）が，それぞれ（a, b），（c, d），（p, q），（r, s），（x, y），（x', y'）に対応する。

3．4　クリッピング

3．4．1　Sutherland-Cohenの方法

線分をクリッピングする方法はSutherlandとCohenの考案したアルゴリズムを用いることにする。これは次のような方法である。

図17に示すように4つの直線で囲まれたスクリーン上のビューポートの範囲を$a \leqq x \leqq c$，$b \leqq y \leqq d$（左手座標系で表現）とし，点（x, y）がどの領域に存在するかを4ビットで表現されたコードで示す。そして，そのビット列の左から右へ第1ビット，第2ビット，……，第4ビットとし，

第1ビット……$x < a$ならば1，
　　そうでなければ0
第2ビット……$x > d$ならば1，
　　そうでなければ0
第3ビット……$x > c$ならば1，
　　そうでなければ0
第4ビット……$x < b$ならば1，
　　そうでなければ0

というビットコードを定める。

そして，そのクリッピングの判定は次のように行う。

①完全可視……両端点の4ビットコードがすべて0の時
②完全可視……両端点の4ビットコードの同一ビットが共に1の時
③クリッピングの候補…上記①，②以外の時（この場合，ビューポートの境界線との交点の情報が必要である）

Sutherlandらのクリッピング・アルゴリズ

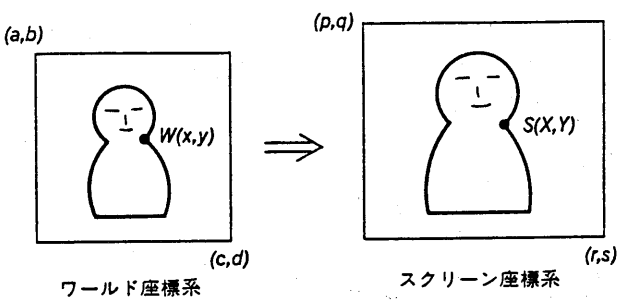


図16　座標交換の例

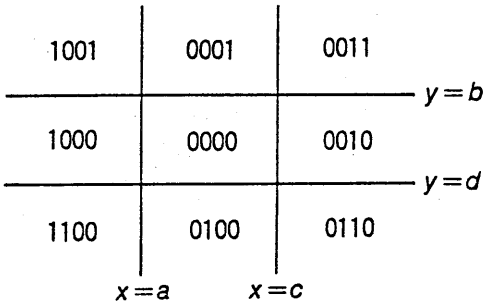


図17　4ビットコード

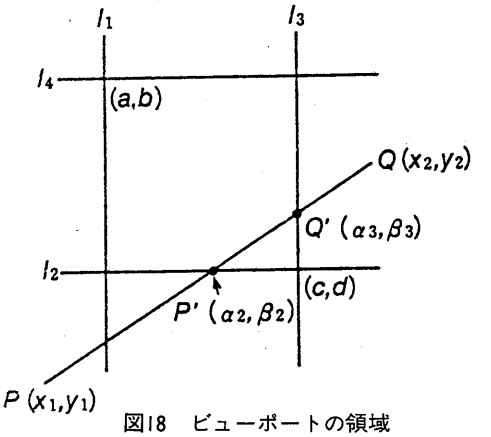


図18　ビューポートの領域

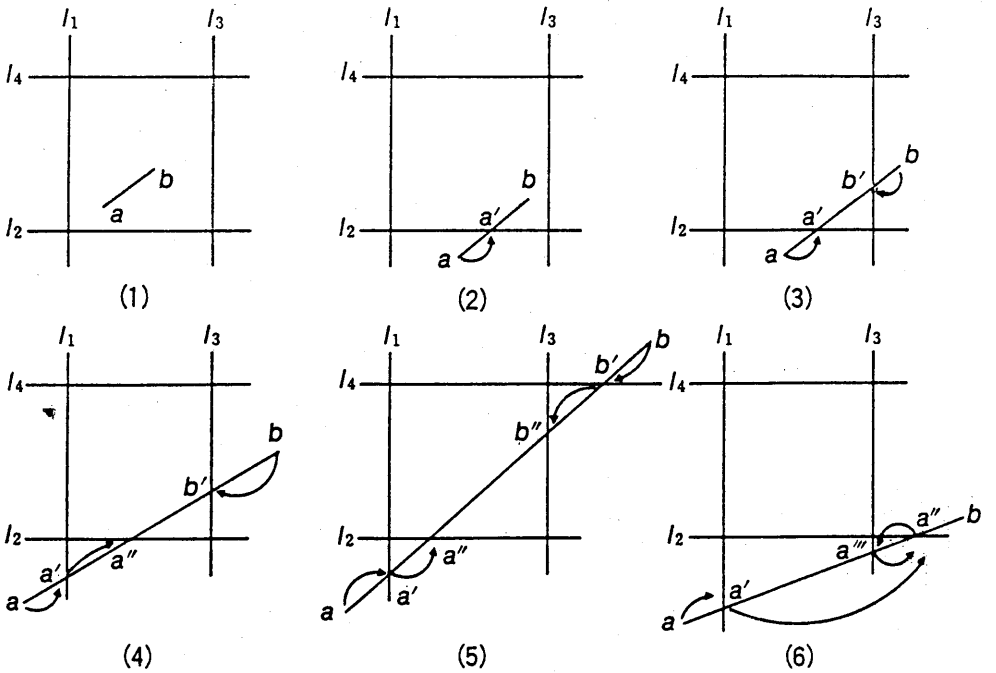


図19　クリッピングの手続き

ムは，ビューポートの境界線を構成する4つの直線と線分の交点を求め，その交点の端点とする線分の4ビットコードがすべて0になるまで，繰り返しながら同様の点を求めていく方法である。

なお，線分とビューポートの境界線との交点は，3．2．1項で示したのと同様な直線の方程式をたてることによって求めることができる。その交点を図18に示すように（α, β）とし，これを次項で用いる。

3．4．2　具体的手続き

クリッピングの具体的手続きを6つのタイプに整理し，以下に図19とともに示す（図中の4本の1はビューポートの境界線である）。

(1)タイプ1（図19(1)の場合）

線分abの端点a，bのそれぞれの4ビットコードを調べ，すべて0であるので完全可視となる。クリッピングされた線分はabそのものである。

(2)タイプ2（図19(2)の場合）

線分abの端点aの4ビットコードは(0100)で，第2ビットが0ではないので部分可視となる。そこでl_2と線分abとの交点a′（α_2, β_2）を求めると，新しい線分a′bの4ビットコードはすべて0になり完全可視となり，クリッピングされた線分はa′bである。

(3)タイプ3（図19(3)）の場合）

線分abの端点（a，b）はそれぞれ(0100)，(0010)で，第2，第3ビットが0ではない。これは部分可視か完全不可視かである。まず線分abとl_2との交点a′（α_2, β_2）を求める。すると線分a′bの端点a′の4ビットコードの第2ビットは0となる。また線分a′bとl_3との交点b′（α_3, β_3）を求めると，線分a′b′の端点の4ビットコードの第3ビットは0となる。そうすると新しい線分a′b′の4ビットコードはすべて0となり，完全可視となる。クリッピングされた線分はa′b′である。

(4)タイプ4（図19(4)の場合）

線分abの端点aの4ビットコードは(1100)である。まず第1ビットが1なので，線分abとl_1との交点a′（α_1, β_1）を求めると，線分a′bの端点a′の4ビットコードは(0100)となる。第2ビットはまだ1なので，さらに線分a′bとl_2との交点a″（α_2, β_2）を求める。すると線分a″bの端点a″の4ビットコードはすべて0となる。線分abの端点aはa→a′→a″へとクリッピングされていく。同様にして端点bもクリッピングし，4ビットコードが0となる端点b′を求める。クリッピングされた線分はa″b′である。

(5)タイプ5（図19(5)の場合）

線分abの端点a，bの4ビットコードは(1100)，(0011)である。それぞれの端点について，4ビットコードがすべて0になるまで，タイプ4で行ったのと同様な手続きで端点を求める。クリッピングされた線分はa″b″である。

(6)タイプ6（図19(6)の場合）

線分abについて，まず端点aについて調べる。その4ビットコードは(1100)である。そこで線分abとl_1との交点a′（α_1, β_1）を求めると，線分a′bの端点a′の4ビットコードは(0100)となる。そこで次に線分a′bとl_2の交点a″（α_2, β_2）を求めると，線分a″bの端点a″の4ビットコードは(0010)となる。そこで今度は線分a″bとl_3の交点a‴（α_3, β_3）を求めると，線分a″bの端点a‴の4ビットコードは(0100)となる。そこでまた線分a‴bとl_2との交点a″（α_2, β_2）を求める。この場合，このようにして求められた線分abの端点はa→a′→a″→a‴→……と繰り返されて，その端点の4ビットコードは決して0とならない。2回の操作で4ビットコードがすべて0とならなければ完全不可視と判定できる。端点bについて同様な手続きを行っても，同じ結果が得られる。

*　　*　　*

上記の6つのタイプを整理して図20に流れ図として示す。図中のnは交点を求めた回数，fは完全不可視かどうかを示すフラッグ（f＝1ならば完全不可視）である。この流れ図を通過すると，線分の新しい端点の座標（α, β）と端点の情報（f）が求まる。

リスト4と図21にSutherlandらのアルゴリズムに基づいて作成されたクリッピングの基礎ルーチンCLIPとその流れ図を示す。ソフトウェア命令であるルーチンCLIPは内部に2つの補助ルーチンGROTとGBITを持っており，GROTは図20の流れ図を実現したものであり，GBITは4ビットコードの設定を

行っている。以下にサブルーチンCLIPの引数とその内容について示す。

・入力用引数

変数AA, BB……クリッピング領域の左上隅の座標

変数CC, DD……クリッピング領域の右下隅の座標

変数X1, Y1……線分abの端点aの座標

変数X2, Y2……線分abの端点bの座標

・出力用引数およびその内容

変数SX, SY……クリッピングされた新しい線分a′b′の端点の座標

変数FLG……可視か不可視かの情報

0：可視

1：不可視

3．5　幾何変換

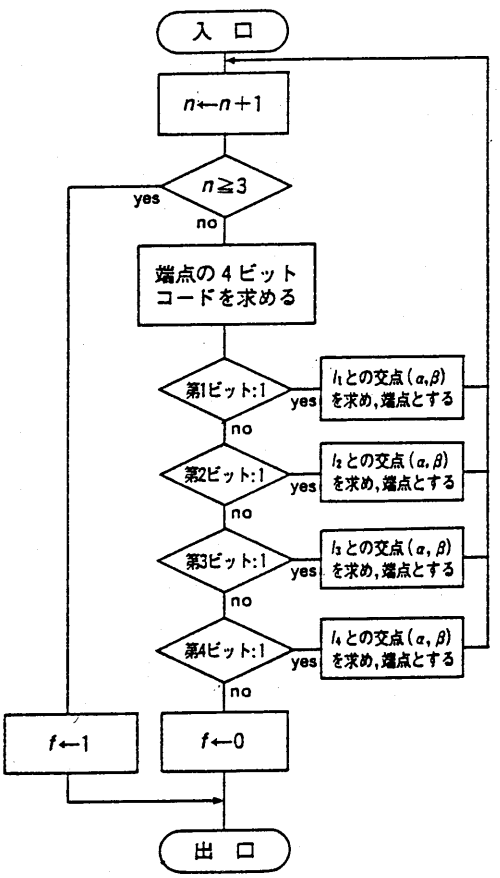


図20　端点のクリッピング

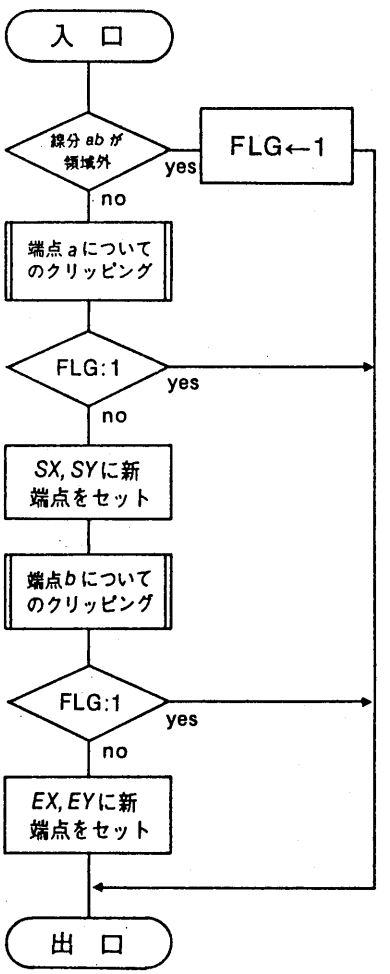


図21　CLIPサブルーチンの流れ図

リスト3

```
4000 '*************************************************************
4010 '*                        CLIP ｻﾌﾞﾙｰﾁﾝ                          *
4020 '*************************************************************
4030 *CLIP
4040 IF X1<AA AND X2<AA THEN FLG=1:GOTO 4150
4050 IF X1>CC AND X2>CC THEN FLG=1:GOTO 4150
4060 IF Y1<BB AND Y2<BB THEN FLG=1:GOTO 4150
4070 IF X1>DD AND Y2>DD THEN FLG=1:GOTO 4150
4080 '
4090 XX=X1:YY=Y1
4100 GOSUB *GROT
4110 IF FLG=0 THEN SX=XX :SY=YY ELSE GOTO 4150
4120 XX=X2:YY=Y2
4130 GOSUB *GROT
4140 IF FLG=0 THEN EX=XX :EY=YY
4150 RETURN
4160 '-------------------------------------------------------------
4170 *GROT
4180 NN=0
4190 NN=NN+1
4200 IF NN>=3 THEN   GOTO 4370
4210 GOSUB *GBIT
4220 IF X1=X2 THEN   GOTO 4300
4230 IF Y1=Y2 THEN   GOTO 4330
4240 IF BT(1)=1 THEN YY=(Y2-Y1)/(X2-X1)*(AA-X1)+Y1:XX=AA:
                     GOTO 4190
4250 IF BT(2)=1 THEN XX=(X2-X1)/(Y2-Y1)*(DD-Y1)+X1:YY=DD:
                     GOTO 4190
4260 IF BT(3)=1 THEN YY=(Y2-Y1)/(X2-X1)*(CC-X1)+Y1:XX=CC:
                     GOTO 4190
4270 IF BT(4)=1 THEN XX=(X2-X1)/(Y2-Y1)*(BB-Y1)+X1:YY=BB:
                     GOTO 4190
4280 GOTO 4360
4290 '
4300 IF BT(2)=1 THEN YY=DD:      GOTO 4190
4310 IF BT(4)=1 THEN YY=BB:      GOTO 4190
4320 IF BT(1)=0 AND BT(3)=0 THEN GOTO 4360 ELSE GOTO 4370
4330 IF BT(1)=1 THEN XX=AA:      GOTO 4190
4340 IF BT(3)=1 THEN XX=CC:      GOTO 4190
4350 IF BT(2)=0 AND BT(4)=0 THEN GOTO 4360 ELSE GOTO 4370
4360 FLG=0:RETURN
4370 FLG=1:RETURN
4380 '-------------------------------------------------------------
4390 *GBIT
4400 IF XX<AA THEN BT(1)=1 ELSE BT(1)=0
4410 IF YY>DD THEN BT(2)=1 ELSE BT(2)=0
4420 IF XX>CC THEN BT(3)=1 ELSE BT(3)=0
4430 IF YY<BB THEN BT(4)=1 ELSE BT(4)=0
4440 RETURN
```

3．5．1　変換の理論式

CGにおける幾何変換の数学的基礎はアフィン変換である。2次元図形の変形は平行移動，拡大縮小，反転，せん断，回転があるが，平行移動以外の変換は2×2変換マトリックスを用いる線形変換の範囲で記述可能である。しかし，平行移動を含める場合には，同次座標を含む3×3変換マトリックスを用いるアフィン変換を導入しなければならない。以下にその変換の理論式を示すが，(x, y)，(x', y')はそれぞれ変換前，変換後の座標である。

(1)平行移動

$$[x'\ y'\ 1]=[x\quad y\quad 1]\begin{pmatrix}1&0&0\\0&1&0\\t_x&t_y&1\end{pmatrix}$$

$$=[(x+t_x)\ (y+t_y)\ 1] \tag{19}$$

(t_x, t_y は x 方向，y 方向への移動量)

(2)拡大縮小

$$[x'\ y'\ 1]=[x\quad y\quad 1]\begin{pmatrix}s_x&0&0\\0&s_y&0\\0&0&1\end{pmatrix} \tag{20}$$

$$=[s_x x\quad s_y y\quad 1]$$

(s_x, s_yはx軸，y軸方向への拡大縮小率)

(3)反転

リスト4

```
5000 '*************************************************************
5010 '*                        COCNV ｻﾌﾞﾙｰﾁﾝ                        *
5020 '*************************************************************
5030 *COCNV
5040 XX=X:YY=Y
5050 FOR J=1 TO 5
5060   IF TA(J)=1 THEN TX=TB(J):TY=TC(J):GOSUB *TRANS
5070   IF TA(J)=2 THEN SX=TB(J):SY=TC(J):GOSUB *SCALE
5080   IF TA(J)=3 THEN IX=TB(J):IY=TC(J):GOSUB *INVRS
5090   IF TA(J)=4 THEN DA=TB(J):DB=TC(J):GOSUB *SHEAR
5100   IF TA(J)=5 THEN TH=TB(J):          GOSUB *ROTAT
5110 NEXT J
5120 RETURN
5130 '-------------------------------------------------------------
5140 *TRANS
5150   XX=XX+TX
5160   YY=YY+TY
5170 RETURN
5180 *SCALE
5190   XX=SX*XX
5200   YY=SY*YY
5210 RETURN
5220 *INVRS
5230   XX=IX*XX
5240   YY=IY*YY
5250 RETURN
5260 *SHEAR
5270   XD=XX:YD=YY
5280   XX=XD+DA*YD
5290   YY=DB*XD+YD
5300 RETURN
5310 *ROTAT
5320   AL=(3.1415/180)*TH
5330   XR=XX:YR=YY
5340   XX=COS(AL)*XR-SIN(AL)*YR
5350   YY=SIN(AL)*XR+COS(AL)*YR
5360 RETURN
```

$$[x'\ y'\ 1]=[x\quad y\quad 1]\begin{pmatrix} i_x & 0 & 0 \\ 0 & i_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{21}$$

$$=[i_x x \qquad i_y y \qquad 1]$$

（i_x, i_yはx軸，y軸方向に対する反転係数）

(4)せん断

$$[x'\quad y'\quad 1]=[x\quad y\quad 1]\begin{pmatrix} 1 & d_b & 0 \\ d_a & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{22}$$

$$=[(x+d_a y)\ (y+d_b x)]$$

（d_a, d_bはx軸，y軸方向に対するせん断率）

(4)回転

$$[x'\quad y'\quad 1]=[x\quad y\ 1]\begin{pmatrix} \cos\theta & \sin\theta & 0 \\ -\cos\theta\sin\theta & 0 & \\ 0 & 0 & 1 \end{pmatrix} \tag{23}$$

$$=[(x\cos\theta-y\sin\theta)\ (x\sin\theta+y\cos\theta)\ 1]$$

（θは原点を中心として反時計回りの回転角度）

3．5．2　合成変換

各種の幾何変換の合成は，個々の変換マトリックスの積から合成変換マトリックスを導き出し，それを座標データに作用させることにより行うことができる。例えば平行移動と回転の個々の変換マトリックスをそれぞれR，Tとし，その合成変換のマトリックスをCとすると，回転後の平行移動の合成変換マトリックスC_{RT}は

$$C_{RT}=RT \tag{24}$$

となる。

一般のCGでは，合成変換はマトリックス演算として処理されるが，BASICにはマトリックス演算機能が備っていない。そこで，幾何変換サブルーチンは，個々の幾何変換の代数式（各変換マトリックスを座標データの列マトリックスに作用させた結果の列マトリックスの要素）を導出し，それを用いて作成することとした。リスト4に作成された幾何変換のソフトウェア命令である基礎ルーチンCOCNVを示す。

サブルーチンCOCNVは大きさが5の配列を用意することにより，最大5つまでの幾何変換を連続して合成することができる。配列TAには幾何変換の種類を以下の番号で指定する。

1：平行移動　　2：拡大縮小
3：反転　　4：せん断
5：回転

合成変換の順は配列TAの要素番号の順である。配列TBには要素番号に対応する各変換のx軸に関する情報，配列TCには同じくy軸に関する情報をセットする（ただし回転の場合のみTBに角度をセットするだけでよい）。

以下にサブルーチンCOCNVの引数とその内容について示す。

・入力用引数
　変数X, Y……合成変換を行う座標
　配列TA……変換の種類
　配列TB……変換のx軸に関する情報
　配列TC……変換のy軸に関する情報

・出力用引数
　変数XX, YY……合成変換された座標

3．6　インターフェース・ルーチン

3．6．1　基礎作画ルーチンの統合

本稿では，これまで次の基礎作画ルーチン等をソフトウェア命令として作成した。

・直線の作画……リスト1
・色塗り……リスト2
・ウィンドウービューポート変換
　…… 3．3 節の定義関数
・クリッピング……リスト3

本節では上記の基礎作画ルーチンを統合化し，図形を描く基本機能をすべて〝自前〟のソフトウェア命令でまかなうことを試みる。この目的が達成されると，ピクセルを発光させる唯一のハードウェア命令であるPSET文だけで，それ以外のグラフィック文を用い

ることなく，どのような図形でも描くことが可能となる。

統合化される個々のサブルーチン等はBASICのグラフィック文と同等の機能を持っているが，それは完全でなかったり，仕様が少し異なっていたりする。そこでBASICのステートメントの機能の仕様になるべく合わせるように，これらのプログラムを改良することにする。〝自前〟のソフトウェア命令のサブルーチンをBASICに対応するハードウェア命令のステートメントと区別するため，表1のように，その名前の先頭にアルファベットの〝U〟を付け，それを以後，ユーザ・サブルーチンと呼ぶ。

ユーザ・サブルーチンのボディ部分はすでに完成しているので，その手直しは，新しいユーザ・サブルーチン名を持ち，その名前を介してボディ部分に橋渡しするインターフェース・ルーチンを作成することである。なお，直線作画のサブルーチンU LINEは，サブルーチンU POINTと組み合せて用いる仕様とし，そのインターフェースを設定する。

3.6.2 インターフェーヌ・ルーチン

リスト5にインターフェース・ルーチンを示す。ユーザ・サブルーチンU WINDOW, U VIEW, U POINT, U LINE, U PAINTは，このルーチンを介して，各サブルーチンのボディの部分（基礎作画ルーチン）

- DDAルーチン
- XPAINTルーチン
- CLIPルーチン

にリンケージする。以下にインターフェース・ルーチンの内容を簡単に明明しておく。

(1) U WINDOWサブルーチン

ウィンドウの領域を定義し，ウィンドウービューポート変換のための定義関数を定め，それを実行する。変数WSWはこのサブルーチンが何度呼び出されてもよいように，スイッチとして働く。

(2) U VIEWサブルーチン

ビューポートの領域の値を変数AA, BB, CC, DDにセットし，それをCLIPルーチンで用いられるようにする。

(3) U POINTサブルーチン

ウィンドウ-ビューポート変換したスクリーン座標の値を変数WKX, WKYに保存する。これはU LINEサブルーチンとの連係に対応するためである。

(4) U LINEサブルーチン

変数WKX, WKYから変数X1, Y1に移されたスクリーン座標での端点と，スクリーン座標に変換されたX2, Y2の端点の座標を用いて，CLIPルーチンを介してクリッピングする。そして線に関するDDAルーチンにより線分を描画する。なお変数WKX, WKYに変数X2, Y2の値を保持す

表1　ユーザ・サブルーチンの仕様

サブルーチン名	引数となる変数・配列	セットする図形情報
UWINDOW	WX1, WY1 WX2, WY2	ウィンドウの左上隅のワールド座標 ウィンドウの右下隅のワールド座標
UVIEW	SX1, SY1 SX2, SY2	ビューポートの左上隅のスクリーン座標 ビューポートの右下隅のスクリーン座標
UPOINT	X, Y	ペンupで移動させる先のワールド座標
ULINE	X, Y COL	ペンdownで移動させる先のワールド座標 座標線の色コード
UPAINT	N PX(n), PY(n) X, Y	境界線分を構成する点の個数 各点のワールド座標 領域内の1点のワールド座標

リスト5

```
6000 '*******************************************************
6010 '*                    UWINDOW ｻﾌﾞﾙｰﾁﾝ                    *
6020 '*******************************************************
6030 *UWINDOW
6040 IF WSW<>0 THEN GOTO 6080
6050 DEF FNX(X)=
          ((SX1-SX2)/(WX1-WX2))*X+(WX1*SX2-SX1*WX2)/(WX1-WX2)
6060 DEF FNY(Y)=
          ((SY1-SY2)/(WY1-WY2))*Y+(WY1*SY2-SY1*WY2)/(WY1-WY2)
6070 WSW=1
6080 RETURN
6090 '*******************************************************
6100 '*                    UVIEW ｻﾌﾞﾙｰﾁﾝ                     *
6110 '*******************************************************
6120 *UVIEW
6130 AA=SX1:BB=SY1:CC=SX2:DD=SY2
6140 RETURN
6150 '*******************************************************
6160 '*                    UPOINT ｻﾌﾞﾙｰﾁﾝ                    *
6170 '*******************************************************
6180 *UPOINT
6190 WKX=FNX(X):WKY=FNY(Y)
6200 RETURN
6210 '*******************************************************
6220 '*                    ULINE ｻﾌﾞﾙｰﾁﾝ                     *
6230 '*******************************************************
6240 *ULINE
6250 X1=WKX:Y1=WKY
6260 X2=FNX(X):Y2=FNY(Y)
6270 WKX=X2:WKY=Y2
6280 GOSUB *CLIP
6290 X1=SX:Y1=SY:X2=EX:Y2=EY
6300 IF FLG=0 THEN GOSUB *DDA
6310 RETURN
6320 '*******************************************************
6330 '*                    UPAINT ｻﾌﾞﾙｰﾁﾝ                    *
6340 '*******************************************************
6350 *UPAINT
6360 GA=(WY2-WY1)/400-1
6370 GOSUB *XPAINT
6380 RETURN
6390 '*******************************************************
6400 '*                    PCTUR ｻﾌﾞﾙｰﾁﾝ                     *
6410 '*******************************************************
6420 *PCTUR
6430 RESTORE
6440 READ GFLG,X,Y,COL
6450   GOSUB *COCNV:X=XX:Y=YY
6460   IF GFLG=0 THEN GOTO 6510
6470   IF GFLG=1 THEN GOSUB *UPOINT:N=0:N=N+1:PX(N)=X:PY(N)=Y
6480   IF GFLG=2 THEN GOSUB *ULINE :N=N+1:    PX(N)=X:PY(N)=Y
6490   IF GFLG=3 THEN GOSUB *UPAINT:N=0
6500   GOTO 6440
6510 RETURN
```

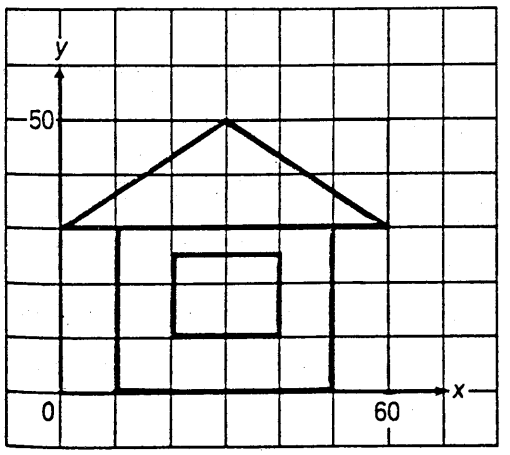


図22　家の図形

るが，これは現在の線分の端点の座標を，後のステップのU LINEサブルーチンで利用できるようにするためである。

(5) U PAINT サブルーチン

ウィンドウービューポート変換を行うことにより，色塗りのための水平直線の移動間隔が問題となる。そこでボディとなるXPAINT ルーチンの変数GA(初期値はゼロ）をこのインターフェース・ルーチンで次のように補正する。

GA=(WY2−WY1) / 400−1

また，それにともないXPAINT ルーチンの行番号3260のLINE文を次の文で置き換える。

```
3260 X=QX(I): Y=QY(I):
     GOSUB  *UPOINT:
     X=QX(I+1): Y=QY(I+1):
     GOSUB  *ULINE
```

4．図形データの定義と描画

4．1　図形のデータの定義

まず，図形の描き方を具体的に例をあげて説明する。例えば図22に示す〝家〟の図形では，まず始めにそれを方眼紙に描く。その座標系は利用者が自由に設定できる右手系のワールド座標系である。座標の範囲は左下隅を（0，0），右上隅を（60，50）とする。この領域に描いた図形の座標は，次に示す一筆書き方式のもとでBASICのDATA文に保存する。これを図23に示す屋根の部分を例にとって説明する。

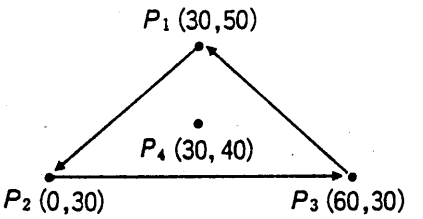


図23　屋根の座標

屋根を描くことは，屋根の輪郭を引き，色を塗ることである。輪郭を引く開始点を屋根の頂点とすれば，まずペンを持ち上げ(up)，点P_1へ移動させる。これはU POINTサブルーチンで行うことができる。次にペンを降して(down)，点$P_1 \rightarrow P_2 \rightarrow P_3$の順に一筆書きで線を引く。これはU LINEサブルーチンで行うことができる。色塗りは点P_4の座標をU PAINTサブルーチンに与えて行う。これで屋根が完成する。

この操作において必要な図形データは，線を引くことに関しては，

- ペンをupに移動かdownに移動かの情報
- ペン移動先の座標データ
- ペンで描くときの線の色コード

であり，色塗りに関しては，

- 色塗り指示のための情報
- 色塗りのための座標データ
- 領域内を塗る色コード

である。したがってこの2種類のデータを次のように整理してDATA文に保存する。

DATA　機能コード，x座標，y座標，色コード

ここで〝機能コード〟とは，次の通りである。

リスト6

```
7000 '*************************************************************
7010 '*                        PCTUR ｻﾌﾞﾙｰﾁﾝ                        *
7020 '*************************************************************
7030 *PCTUR
7040 RESTORE
7050 READ GFLG,X,Y,COL
7060   GOSUB *COCNV:X=XX:Y=YY
7070   IF GFLG=0 THEN GOTO 7120
7080   IF GFLG=1 THEN GOSUB *UPOINT:N=0:N=N+1:PX(N)=X:PY(N)=Y
7090   IF GFLG=2 THEN GOSUB *ULINE :N=N+1:    PX(N)=X:PY(N)=Y
7100   IF GFLG=3 THEN GOSUB *UPAINT:N=0
7110   GOTO 7050
7120 RETURN
```

表2　PCTURサブルーチンの仕様

サブルーチン名	引数となる変数・配列	セットする図形情報
PCTUR	●COCNVサブルーチンで用いられる配列を使用	
	TA(1)～TA(5)	変換の種類
	TB(1)～TB(5)	変換のx軸に関する情報
	TC(1)～TC(5)	変換のy軸に関する情報
	●DATA文を使用	
	1番目	機能コード
	2番目	図形のワールドx座標
	3番目	図形のワールドy座標
	4番目	線の色コード

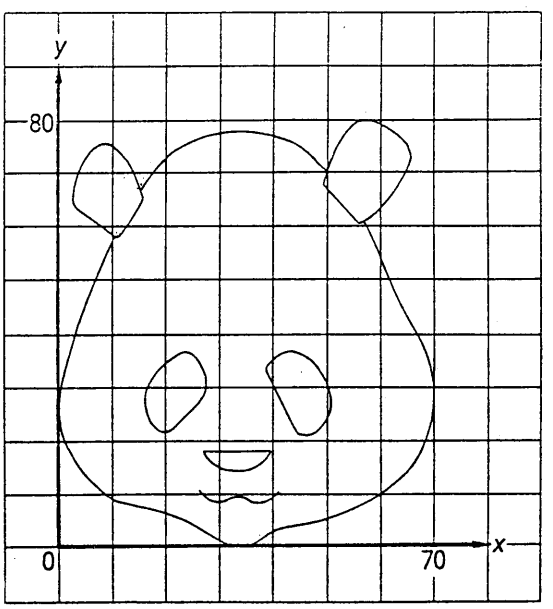


図24　パンダの原図

リスト7

```
8000 '*************************************************************
8010 '*                        ｽﾞｹｲ ﾃﾞｰﾀ                            *
8020 '*************************************************************
8030 '------------------------------------------  ﾘﾝｶｸ  -----
8040 DATA   1,30,00,7,    2,36,00,7,    2,38,02,7
8050 DATA   2,40,03,7,    2,54,07,7,    2,66,16,7
8060 DATA   2,70,24,7,    2,70,30,7,    2,63,43,7
8070 DATA   2,60,50,7,    2,56,60,7,    2,49,69,7
8080 DATA   2,42,75,7,    2,33,77,7,    2,26,75,7
8090 DATA   2,14,67,7,    2,09,57,7,    2,03,41,7
8100 DATA   2,00,23,7,    2,03,14,7,    2,12,07,7
8110 DATA   2,23,05,7,    2,25,04,7,    2,30,00,7
8120 DATA   3,30,20,7
8130 '------------------------------------------  ﾋﾀﾞﾘ ﾐﾐ  --
8140 DATA   1,56,60,0,    2,63,70,0,    2,64,72,0
8150 DATA   2,63,75,0,    2,59,80,0,    2,53,78,0
8160 DATA   2,49,69,0,    2,48,67,0,    2,53,60,0
8170 DATA   2,56,60,0
8180 DATA   3,60,70,0
8190 '------------------------------------------  ﾐｷﾞ ﾐﾐ  ---
8200 DATA   1,15,64,0,    2,14,67,0,    2,09,75,0
8210 DATA   2,06,74,0,    2,02,67,0,    2,02,63,0
8220 DATA   2,09,57,0,    2,11,57,0,    2,15,64,0
8230 DATA   3,10,60,0
8240 '------------------------------------------  ﾋﾀﾞﾘ ﾒ  ---
8250 DATA   1,44,21,0,    2,46,20,0,    2,49,22,0
8260 DATA   2,50,27,0,    2,47,33,0,    2,43,36,0
8270 DATA   2,39,34,0,    2,38,31,0,    2,44,21,0
8280 DATA   3,40,30,0
8290 '------------------------------------------  ﾐｷﾞ ﾒ  ----
8300 DATA   1,22,21,0,    2,26,27,0,    2,27,31,0
8310 DATA   2,24,36,0,    2,18,32,0,    2,15,27,0
8320 DATA   2,19,21,0,    2,22,21,0
8330 DATA   3,20,30,0
8340 '------------------------------------------  ﾊﾅ  -------
8350 DATA   1,26,18,0,    2,29,14,0,    2,32,13,0
8360 DATA   2,36,15,0,    2,39,18,0,    2,26,18,0
8370 DATA   3,35,16,0
8380 '------------------------------------------  ｸﾁ  -------
8390 DATA   1,25,10,0,    2,27,08,0,    2,29,08,0
8400 DATA   2,31,09,0,    2,33,09,0,    2,36,08,0
8410 DATA   2,38,09,0,    2,40,10,0
8420 DATA   0,00,00,0
```

リスト8

```
1000 '*************************************************************
1010 '*                        ﾒｲﾝ ﾌﾟﾛｸﾞﾗﾑ                           *
1020 '*************************************************************
1030 SCREEN 3,0:CONSOLE ,,0,1:CLS 3
1040 WX1=-175:WY1= 200:WX2= 175:WY2=-200: GOSUB *UWINDOW
1050 SX1= 144:SY1=   0:SX2= 494:SY2= 399: GOSUB *UVIEW
```

```
1060 DIM PX(100),PY(100)
1070 N=5:X=0:Y=0:COL=2
1080 PX(1)=WX1   :PY(1)=WY1
1090 PX(2)=WX1   :PY(2)=WY2
1100 PX(3)=WX2   :PY(3)=WY2
1110 PX(4)=WX2   :PY(4)=WY1
1120 PX(5)=WX1   :PY(5)=WY1
1130 GOSUB *UPAINT
1140 '-----------------------------------------------------
1150 TA(1)=1 :TB(1)= -35 :TC(1)= -40
1160 TA(2)=5 :TB(2)= 330 :TC(2)=   0
1170 TA(3)=2 :TB(3)=   1 :TC(3)=   1
1180 TA(4)=1 :TB(4)= 150 :TC(4)= 105
1190 GOSUB *PCTUR
1200 TA(1)=1 :TB(1)= -35 :TC(1)= -40
1210 TA(2)=5 :TB(2)=  30 :TC(2)=   0
1220 TA(3)=2 :TB(3)=   1 :TC(3)=   1
1230 TA(4)=1 :TB(4)=-150 :TC(4)= 105
1240 GOSUB *PCTUR
1250 TA(1)=1 :TB(1)= -35 :TC(1)= -40
1260 TA(2)=5 :TB(2)=   0 :TC(2)=   0
1270 TA(3)=2 :TB(3)= 2.6 :TC(3)= 2.6
1280 TA(4)=1 :TB(4)=   0 :TC(4)= -65
1290 GOSUB *PCTUR
1300 END
```

1：ペン up で移動

2：ペン down で移動

3：色を塗る

上記の規則を利用して，屋根の図形データは次のように定義される。

```
DATA  1, 30, 50,  0
DATA  2,  0, 30,  2
DATA  2, 60, 30,  2
DATA  2, 30, 50,  2
DATA  3, 30, 40,  2
```

この方式を用いると，原理的にどのような図形でも描くことが可能となる。

4．2　図形描画サブルーチン

図形データを読み出して描画するサブルーチンは次のように作成する。READ 文で読み出されたデータは機能コードによって，U POINT, U LINE, U PAINT の各サブルーチンに振り分けるようにする。機能コードが 0 であれば，図形データの読み出しは中止する。

このようなサブルーチンをリスト 6 に示す。サブルーチン PCTUR の行番号6060に記述された COCNV ルーチンはリスト 4 に示した幾何変換サブルーチンである。幾何変換は READ で読み出された 1 件 1 件の図形データに対して行われる。なお表 2 にサブルーチン PCTUR の引数について示す。

5．おわりに……パンダの描画

サブルーチン PCTUR を用いて自由な図形を描いてみる。このためには，これまでに作成した基礎作画ルーチンとインターフェース・ルーチンをすべて必要とする。描く図形は図24[7]に示すパンダの顔である。この図から73点の座標を読み取り，それを図形データとしてDATA文に登録する。それをリスト 7 に示す。

パンダを描くメインプログラムをリスト 8 に示す。行番号1040と1050でウィンドウとビューポートを定義し，行番号1130でビューポ

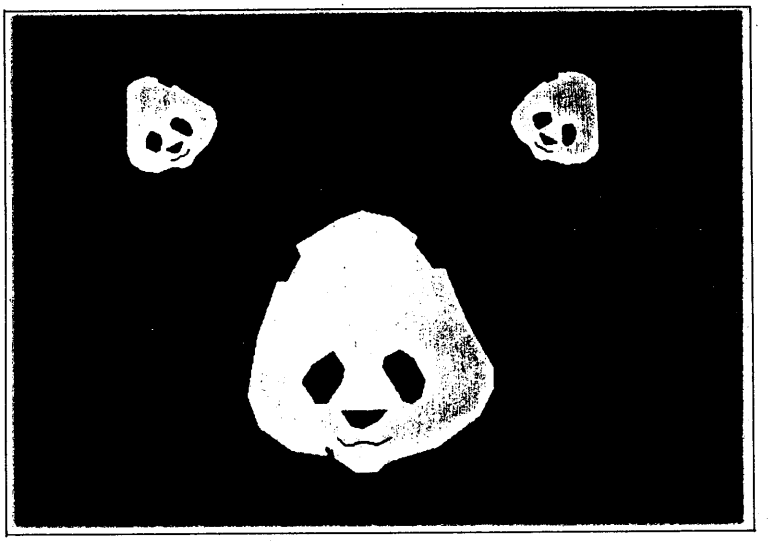


図25　パンダの出力

ート領域を赤色とし，行番号1190，1240，1290でパンダの顔を3つ描く。その結果を図25の写真に示す。3つのパンダの顔のうち2つは拡大率が異なり，上部の2つの小さな顔は，それぞれ左右30°回転している。またその2つの顔はビューポートの境界でクリッピングされている。

このパンダの図形データはサブルーチンPCTURから処理され始め，ソフトウェア命令としてのインターフェース・ルーチンおよび基礎作画ルーチンを経由しながら，

・ウィンドウービューポート変換
・クリッピング
・幾何変換

等の操作が行われ，最終的には唯一のハードウェア命令であるPSET文によるコントロールのもとで，ピクセル（点）として発光して図形として現れるのである。この過程を図26の流れ図に示す。このことからラスターグラフィックにおいては，「図形描画の基本は点である」という基本原理を改めて認識することができる。

なお，最後に付録として，本稿で分割して示したプログラムをまとめて掲載しておく。

＊　　＊　　＊

本稿は拙書[1]に掲載した内容の一部分を論文形式に要約・整理して，昭和61年度文教大学情報学部共同研究の報告としたものである。

内容の部分的利用を御了解下さった啓学出版の野沢裕氏，および本稿の紀要への掲載を御許可下さった同大学情報学部紀要員会に感謝致します。

参考文献

(1) 広内哲夫著『コンピュータ・グラフィックス—CG理論の展開と応用—』，啓学出版，1987。
(2) D. F. ロジャーズ，山口富士夫監修，セイコー電子工業訳『実践コンピュータ・グラフィックス—基礎手続きと応用—』，日刊工業新聞社，1987。
(3) R.A.プラストック，G.カレイ著，郡山彬訳『コンピュータグラフィックス』，マグロウヒルブック，1987。
(4) 佐藤義雄著『入門グラフィックス』，アスキー出版局，1984。
(5) 山口富士夫著『コンピュータディスプレイによる図形処理工学』，日刊工業新聞社，1981。
(6) 広内哲夫著『プログラミング言語BASIC』，啓学出版，1987。
(7) パンダの図形は，山下勇三氏制作の『上野動物園パンダの名前募集ポスター』の中の図柄の一部を，同氏の御厚意により利用させて頂いた。

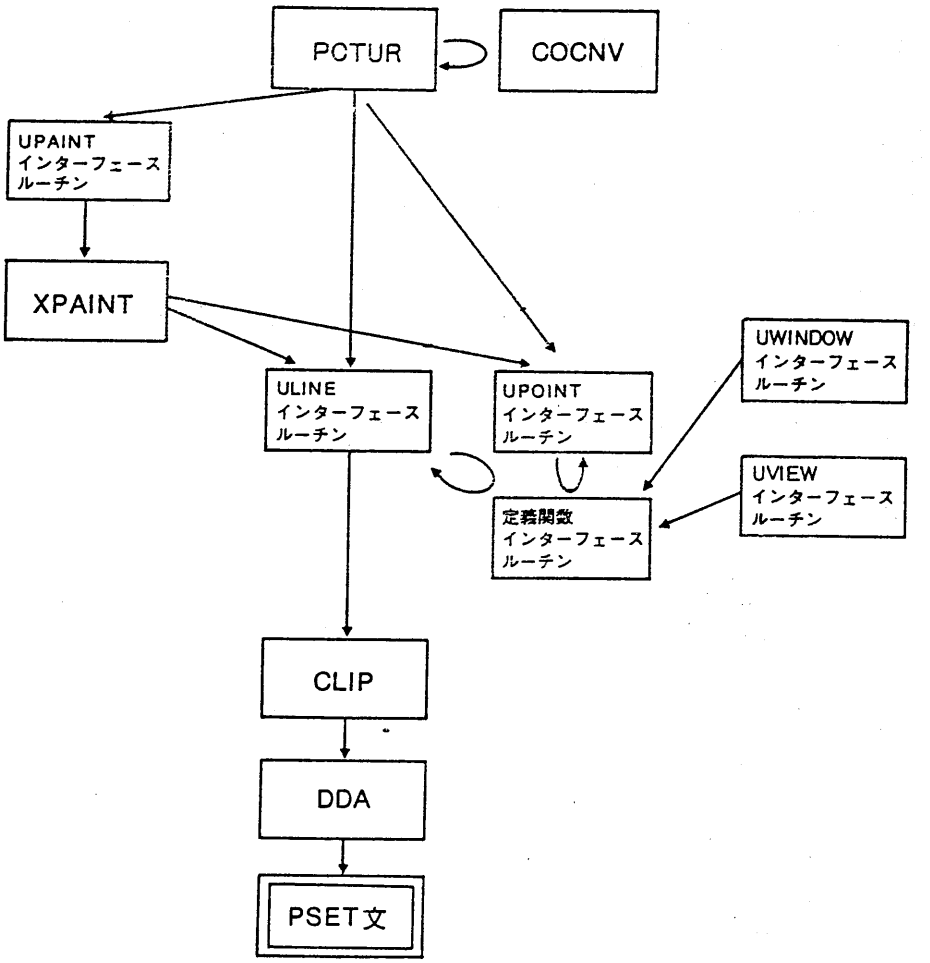


図26　サブルーチン間の関係

付録

```
1000 '************************************************************
1010 '*                          ﾒｲﾝ ﾌﾟﾛｸﾞﾗﾑ                      *
1020 '************************************************************
1030 SCREEN 3,0:CONSOLE ,,0,1:CLS 3
1040 WX1=-175:WY1= 200:WX2= 175:WY2=-200: GOSUB *UWINDOW
1050 SX1= 144:SY1=   0:SX2= 494:SY2= 399: GOSUB *UVIEW
1060 DIM PX(100),PY(100)
1070 N=5:X=0:Y=0:COL=2
1080 PX(1)=WX1   :PY(1)=WY1
1090 PX(2)=WX1   :PY(2)=WY2
1100 PX(3)=WX2   :PY(3)=WY2
1110 PX(4)=WX2   :PY(4)=WY1
1120 PX(5)=WX1   :PY(5)=WY1
1130 GOSUB *UPAINT
1140 '------------------------------------------------------------
```

```
1150 TA(1)=1 :TB(1)= -35 :TC(1)= -40
1160 TA(2)=5 :TB(2)= 330 :TC(2)=   0
1170 TA(3)=2 :TB(3)=   1 :TC(3)=   1
1180 TA(4)=1 :TB(4)= 150 :TC(4)= 105
1190 GOSUB *PCTUR
1200 TA(1)=1 :TB(1)= -35 :TC(1)= -40
1210 TA(2)=5 :TB(2)=  30 :TC(2)=   0
1220 TA(3)=2 :TB(3)=   1 :TC(3)=   1
1230 TA(4)=1 :TB(4)=-150 :TC(4)= 105
1240 GOSUB *PCTUR
1250 TA(1)=1 :TB(1)= -35 :TC(1)= -40
1260 TA(2)=5 :TB(2)=   0 :TC(2)=   0
1270 TA(3)=2 :TB(3)= 2.6 :TC(3)= 2.6
1280 TA(4)=1 :TB(4)=   0 :TC(4)= -65
1290 GOSUB *PCTUR
1300 END
2000 '*************************************************************
2010 '*                        DDA ｻﾌﾞﾙｰﾁﾝ                          *
2020 '*************************************************************
2030 *DDA
2040 DEFINT I
2050 IX1=X1:IY1=Y1:IX2=X2:IY2=Y2
2060 IDX=IX2-IX1   :IDY=IY2-IY1
2070 IA=ABS(IDX)   :IB=ABS(IDY)
2080 IX=IX1        :IY=IY1
2090 IF IA>=IB THEN GOTO 2110 ELSE GOTO 2190
2100 '-------------------------------------------------------
2110 IE=-ABS(IDX)
2120 PSET(IX,IY),COL
2130   IF IX=IX2 THEN GOTO 2260
2140   IX=IX+SGN(IDX)
2150   IE=IE+2*IB
2160   IF IE>=0 THEN IY=IY+SGN(IDY):IE=IE-2*IA
2170   GOTO 2120
2180 '-------------------------------------------------------
2190 IE=-ABS(IDY)
2200 PSET(IX,IY),COL
2210   IF IY=IY2 THEN GOTO 2260
2220   IY=IY+SGN(IDY)
2230   IE=IE+2*IA
2240   IF IE>=0 THEN IX=IX+SGN(IDX):IE=IE-2*IB
2250   GOTO 2200
2260 DEFSNG I
2270 RETURN
3000 '*************************************************************
3010 '*                      XPAINT ｻﾌﾞﾙｰﾁﾝ                         *
3020 '*************************************************************
3030 *XPAINT
3040 IF PSW=0 THEN DIM QX(20),QY(20):PSW=1
3050 CY=Y
3060 H=CY
3070 GOSUB *GLN
3080   IF FLG=0 THEN H=H+1+GA:GOTO 3070
```

```
3090 H=CY
3100 GOSUB *GLN
3110   IF FLG=0 THEN H=H-1-GA:GOTO 3100
3120 RETURN
3130 '-------------------------------------------------------
3140 *GLN
3150 M=0
3160 FOR I=1 TO N
3170   X1=PX(I):Y1=PY(I)
3180   IF I<>N THEN X2=PX(I+1):Y2=PY(I+1) ELSE X2=PX(1):
                                               Y2=PY(1)
3190   IF Y1<>Y2 THEN X=(H-Y1)*(X2-X1)/(Y2-Y1)+X1:Y=H
                 ELSE GOTO 3220
3200   IF X=>X1 AND X=<X2 OR X=>X2 AND X=<X1 THEN
                                                ELSE GOTO 3220
3210   IF Y=>Y1 AND Y=<Y2 OR Y=>Y2 AND Y=<Y1 THEN GOSUB *GCHK
3220 NEXT I
3230 IF M=0 THEN FLG=1:GOTO 3290
3240 GOSUB *GSRT
3250 FOR I=1 TO M STEP 2
3260   X=QX(I  ):Y=QY(I  ):GOSUB *UPOINT:
       X=QX(I+1):Y=QY(I+1):GOSUB *ULINE
3270 NEXT I
3280 FLG=0
3290 RETURN
3300 '-------------------------------------------------------
3310 *GCHK
3320 IF I=N THEN GOTO 3340
3330 P=M:Q=I-1:R=I+1:GOSUB *GSET:GOTO 3380
3340 IF QX(M)=X THEN GOTO 3360
3350 IF QX(1)=X THEN GOTO 3370
3360 P=M:Q=N-1:R=1  :GOSUB *GSET:GOTO 3380
3370 P=1:Q=N  :R=2  :GOSUB *GSET
3380 RETURN
3390 '-------------------------------------------------------
3400 *GSET
3410 IF M=0 THEN M=M+1:QX(M)=X:QY(M)=Y:GOTO 3430
3420 IF NOT (QX(P)=X AND ((Y>PY(Q) AND Y<PY(R)) OR (Y<PY(Q)
                  AND Y>PY(R)))) THEN M=M+1:QX(M)=X:QY(M)=Y
3430 RETURN
3440 '-------------------------------------------------------
3450 *GSRT
3460 FOR J=1 TO M
3470   VL=QX(J)
3480   ID=J
3490   FOR K=J TO M
3500     IF QX(K)<VL THEN VL=QX(K):ID=K
3510   NEXT K
3520   SWAP QX(J),VL
3530   QX(ID)=VL
3540 NEXT J
3550 RETURN
```

```
4000 '*********************************************************
4010 '*                      CLIP ｻﾌﾞﾙｰﾁﾝ                       *
4020 '*********************************************************
4030 *CLIP
4040 IF X1<AA AND X2<AA THEN FLG=1:GOTO 4150
4050 IF X1>CC AND X2>CC THEN FLG=1:GOTO 4150
4060 IF Y1<BB AND Y2<BB THEN FLG=1:GOTO 4150
4070 IF X1>DD AND Y2>DD THEN FLG=1:GOTO 4150
4080 '
4090 XX=X1:YY=Y1
4100 GOSUB *GROT
4110 IF FLG=0 THEN SX=XX :SY=YY ELSE GOTO 4150
4120 XX=X2:YY=Y2
4130 GOSUB *GROT
4140 IF FLG=0 THEN EX=XX :EY=YY
4150 RETURN
4160 '---------------------------------------------------------
4170 *GROT
4180 NN=0
4190 NN=NN+1
4200 IF NN>=3 THEN    GOTO 4370
4210 GOSUB *GBIT
4220 IF X1=X2 THEN    GOTO 4300
4230 IF Y1=Y2 THEN    GOTO 4330
4240 IF BT(1)=1 THEN YY=(Y2-Y1)/(X2-X1)*(AA-X1)+Y1:XX=AA:
                      GOTO 4190
4250 IF BT(2)=1 THEN XX=(X2-X1)/(Y2-Y1)*(DD-Y1)+X1:YY=DD:
                      GOTO 4190
4260 IF BT(3)=1 THEN YY=(Y2-Y1)/(X2-X1)*(CC-X1)+Y1:XX=CC:
                      GOTO 4190
4270 IF BT(4)=1 THEN XX=(X2-X1)/(Y2-Y1)*(BB-Y1)+X1:YY=BB:
                      GOTO 4190
4280 GOTO 4360
4290 '
4300 IF BT(2)=1 THEN YY=DD:       GOTO 4190
4310 IF BT(4)=1 THEN YY=BB:       GOTO 4190
4320 IF BT(1)=0 AND BT(3)=0 THEN GOTO 4360 ELSE GOTO 4370
4330 IF BT(1)=1 THEN XX=AA:       GOTO 4190
4340 IF BT(3)=1 THEN XX=CC:       GOTO 4190
4350 IF BT(2)=0 AND BT(4)=0 THEN GOTO 4360 ELSE GOTO 4370
4360 FLG=0:RETURN
4370 FLG=1:RETURN
4380 '---------------------------------------------------------
4390 *GBIT
4400 IF XX<AA THEN BT(1)=1 ELSE BT(1)=0
4410 IF YY>DD THEN BT(2)=1 ELSE BT(2)=0
4420 IF XX>CC THEN BT(3)=1 ELSE BT(3)=0
4430 IF YY<BB THEN BT(4)=1 ELSE BT(4)=0
4440 RETURN
5000 '*********************************************************
5010 '*                     COCNV ｻﾌﾞﾙｰﾁﾝ                       *
5020 '*********************************************************
5030 *COCNV
```

```
5040 XX=X:YY=Y
5050 FOR J=1 TO 5
5060   IF TA(J)=1 THEN TX=TB(J):TY=TC(J):GOSUB *TRANS
5070   IF TA(J)=2 THEN SX=TB(J):SY=TC(J):GOSUB *SCALE
5080   IF TA(J)=3 THEN IX=TB(J):IY=TC(J):GOSUB *INVRS
5090   IF TA(J)=4 THEN DA=TB(J):DB=TC(J):GOSUB *SHEAR
5100   IF TA(J)=5 THEN TH=TB(J):          GOSUB *ROTAT
5110 NEXT J
5120 RETURN
5130 '-------------------------------------------------------
5140 *TRANS
5150   XX=XX+TX
5160   YY=YY+TY
5170 RETURN
5180 *SCALE
5190   XX=SX*XX
5200   YY=SY*YY
5210 RETURN
5220 *INVRS
5230   XX=IX*XX
5240   YY=IY*YY
5250 RETURN
5260 *SHEAR
5270   XD=XX:YD=YY
5280   XX=XD+DA*YD
5290   YY=DB*XD+YD
5300 RETURN
5310 *ROTAT
5320   AL=(3.1415/180)*TH
5330   XR=XX:YR=YY
5340   XX=COS(AL)*XR-SIN(AL)*YR
5350   YY=SIN(AL)*XR+COS(AL)*YR
5360 RETURN
6000 '*********************************************************
6010 '*                      UWINDOW ｻﾌﾞﾙｰﾁﾝ                    *
6020 '*********************************************************
6030 *UWINDOW
6040 IF WSW<>0 THEN GOTO 6080
6050 DEF FNX(X)=
        ((SX1-SX2)/(WX1-WX2))*X+(WX1*SX2-SX1*WX2)/(WX1-WX2)
6060 DEF FNY(Y)=
        ((SY1-SY2)/(WY1-WY2))*Y+(WY1*SY2-SY1*WY2)/(WY1-WY2)
6070 WSW=1
6080 RETURN
6090 '*********************************************************
6100 '*                      UVIEW ｻﾌﾞﾙｰﾁﾝ                     *
6110 '*********************************************************
6120 *UVIEW
6130 AA=SX1:BB=SY1:CC=SX2:DD=SY2
6140 RETURN
6150 '*********************************************************
6160 '*                      UPOINT ｻﾌﾞﾙｰﾁﾝ                    *
6170 '*********************************************************
```

```
6180 *UPOINT
6190 WKX=FNX(X):WKY=FNY(Y)
6200 RETURN
6210 '************************************************************
6220 '*                        ULINE ｻﾌﾞﾙｰﾁﾝ                        *
6230 '************************************************************
6240 *ULINE
6250 X1=WKX:Y1=WKY
6260 X2=FNX(X):Y2=FNY(Y)
6270 WKX=X2:WKY=Y2
6280 GOSUB *CLIP
6290 X1=SX:Y1=SY:X2=EX:Y2=EY
6300 IF FLG=0 THEN GOSUB *DDA
6310 RETURN
6320 '************************************************************
6330 '*                        UPAINT ｻﾌﾞﾙｰﾁﾝ                       *
6340 '************************************************************
6350 *UPAINT
6360 GA=(WY2-WY1)/400-1
6370 GOSUB *XPAINT
6380 RETURN
7000 '************************************************************
7010 '*                        PCTUR ｻﾌﾞﾙｰﾁﾝ                        *
7020 '************************************************************
7030 *PCTUR
7040 RESTORE
7050 READ GFLG,X,Y,COL
7060   GOSUB *COCNV:X=XX:Y=YY
7070   IF GFLG=0 THEN GOTO 7120
7080   IF GFLG=1 THEN GOSUB *UPOINT:N=0:N=N+1:PX(N)=X:PY(N)=Y
7090   IF GFLG=2 THEN GOSUB *ULINE :N=N+1:    PX(N)=X:PY(N)=Y
7100   IF GFLG=3 THEN GOSUB *UPAINT:N=0
7110   GOTO 7050
7120 RETURN
8000 '************************************************************
8010 '*                        ｽﾞｹｲ ﾃﾞｰﾀ                           *
8020 '************************************************************
8030 '-------------------------------------------  ﾘﾝｶｸ  -----
8040 DATA   1,30,00,7,    2,36,00,7,    2,38,02,7
8050 DATA   2,40,03,7,    2,54,07,7,    2,66,16,7
8060 DATA   2,70,24,7,    2,70,30,7,    2,63,43,7
8070 DATA   2,60,50,7,    2,56,60,7,    2,49,69,7
8080 DATA   2,42,75,7,    2,33,77,7,    2,26,75,7
8090 DATA   2,14,67,7,    2,09,57,7,    2,03,41,7
8100 DATA   2,00,23,7,    2,03,14,7,    2,12,07,7
8110 DATA   2,23,05,7,    2,25,04,7,    2,30,00,7
8120 DATA   3,30,20,7
8130 '-------------------------------------------  ﾋﾀﾞﾘ ﾐﾐ  --
8140 DATA   1,56,60,0,    2,63,70,0,    2,64,72,0
8150 DATA   2,63,75,0,    2,59,80,0,    2,53,78,0
8160 DATA   2,49,69,0,    2,48,67,0,    2,53,60,0
8170 DATA   2,56,60,0
8180 DATA   3,60,70,0
```

```
8190 '------------------------------------------ ﾐｷﾞ ﾐﾐ ---
8200 DATA   1,15,64,0,    2,14,67,0,    2,09,75,0
8210 DATA   2,06,74,0,    2,02,67,0,    2,02,63,0
8220 DATA   2,09,57,0,    2,11,57,0,    2,15,64,0
8230 DATA   3,10,60,0
8240 '------------------------------------------ ﾋﾀﾞﾘ ﾒ ---
8250 DATA   1,44,21,0,    2,46,20,0,    2,49,22,0
8260 DATA   2,50,27,0,    2,47,33,0,    2,43,36,0
8270 DATA   2,39,34,0,    2,38,31,0,    2,44,21,0
8280 DATA   3,40,30,0
8290 '------------------------------------------ ﾐｷﾞ ﾒ ----
8300 DATA   1,22,21,0,    2,26,27,0,    2,27,31,0
8310 DATA   2,24,36,0,    2,18,32,0,    2,15,27,0
8320 DATA   2,19,21,0,    2,22,21,0
8330 DATA   3,20,30,0
8340 '------------------------------------------ ﾊﾅ -------
8350 DATA   1,26,18,0,    2,29,14,0,    2,32,13,0
8360 DATA   2,36,15,0,    2,39,18,0,    2,26,18,0
8370 DATA   3,35,16,0
8380 '------------------------------------------ ｸﾁ -------
8390 DATA   1,25,10,0,    2,27,08,0,    2,29,08,0
8400 DATA   2,31,09,0,    2,33,09,0,    2,36,08,0
8410 DATA   2,38,09,0,    2,40,10,0
8420 DATA   0,00,00,0
```