パソコン用画像処理プログラム

吉村　卓

Program of Image Processing for Microcomputer

Takashi YOSHIMURA

(昭和60年10月28日受理)

1. はしがき

ディジタル画像のデータ量は非常に膨大で，1画面の画像に対して多くのメモリ量を必要とし，さらに，計算処理に多くの時間を必要とするので，計算機としては大容量かつ高速演算のできる大型計算機が必要とされてきた。

近年，マイクロプロセッサやメモリが安価になりパーソナルコンピュータとして安価な計算機が普及し，パソコンの高機能化とともにその周辺機器の一つとしてパソコン用の画像処理ユニットが開発され，安価に画像処理システムが構成できるようになった。

以下に，ＮＥＣのＰＣ－９８０１Ｆを用いた画像処理システムについて述べる。

2. システムの規模と画像のデータ構造

画像データをディジタル化するためには，画面をいくつかの画素（標本点）に分けて，その標本点における濃度をディジタル化（量子化）しなければならない。標本点の数および量子化のレベル数が多いほどきめ細かい美しい画像が得られるが，それだけ必要とするメモリ量が多くなり，計算処理時間も長くなる。したがって，画素数および量子化のレベルはパソコンのメモリ容量・演算速度を考慮して必要最少限にとどめなければならない。

このシステムでは1枚の画面を表現するのに256＊256画素とし，各画素は4ビット（16段階）の濃度値データをもつものとする。

人間の眼に自然な濃淡を感じさせるためには64レベル（6ビット）程度が必要とされるが，処理の目的によってはそれ以下のレベルでも十分である。

各画素は画像メモリ上0000番地からFFFF番地までそれぞれ16ビットの番地で指定される1バイトに格納される。

メモリの節約をはかる目的から1バイトに1画素4ビットのデータを2画素分格納する詰込み方式もあるが，詰め込まれた形のデータを直接取り扱うことはできないから，各画素を処理する際はいったん1画素／1バイトのデータに分離してから各種演算を施し，処理後は再び2画素を1バイトに合成して画像メモリに格納する必要がある。このシステムでは記憶域へのアクセスの効率を考えて1画素1バイトで表現することとする。

1バンク６４Ｋバイトに1枚の画像（２５６＊２５６画素）を格納することになる。

画素の位置を表わすのに格納されている画像メモリの番地を用いることにすれば，4桁の１６進数で表示される番地の下位2桁が画面上のＸ座標を，上位2桁がｙ座標を表わすことになる。すなわち，左右方向を右向きにｘ軸，上下方向を下向きにｙ軸をとると，画素番地は座標（ｘ，ｙ）で表わされ，画面左上隅は（０，０），右下隅は（ＦＦ，ＦＦ）と表わされる。

画像メモリの画像をフロッピィディスクに記録したり，フロッピィディスクに保存されている画像データを画像メモリにロードするための入出力操作の高速化とメモリの節約を考慮してデータファイルにはランダム　アクセス　ファイルを用いる。ファイル　バッファの大きさは２５６バイトなので，1回の入出力操作で横一線分２５６画素ずつ転送される。

3. 処理機能の概要

以下にこのシステムにおける処理の種類（メニュー）について述べる。

３・１　ＲＩＧＩＯＮ（処理領域の設定）

処理する領域を限定したい場合に用いる。処理領域の設定の仕方として次の三つのサブメニューがある。

(1)　ＡＬＬ　　処理範囲を画面全体すなわち２５６＊２５６画素とする。

(2)　ＭＡＮＵＡＬ　　長方形領域の左上隅の座標と右下隅の座標を入力することにより設定する。

(3) ＣＵＲＳＯＲ　カーソルを動かして処理領域を設定する。カーソルはキーボードのテンキーを用いて動かす。カーソルは処理領域を区切る上方横線と左側縦線，ならびに下方横線と右側縦線がそれぞれ同時に動く。

３・２　ＣＵＲＳＯＲ（カーソルの表示・消去）

カーソルは画面上にカーソル線上の画素の濃度値を反転して描かれるので，この命令を実行するとカーソルが描かれている場合はカーソルが消え，カーソルが消えている場合はカーソルが描かれる。

３・３　ＳＡＶＥ（画像の保存）

画像データをフロッピディスクに格納する。セーブする前に，セーブする範囲をＲＩＧＩＯＮで設定しておく必要がある。

３・４　ＬＯＡＤ（画像の再生）

フロッピディスクに保存しておいた画像データを読み出し，ディスプレイに表示する。

３・５　ＭＯＮＴＡＧＥ（画像の合成）

ディスプレイ画面上にある画像Ａとフロッピディスクに保存してある画像Ｂの２枚の画像の間で合成処理をする。合成するデータ領域として

(1) ＯＲＩＧＩＮＡＬ　セーブしておいた画像をそのままの位置で画面上の画像と合成する。

(2) ＭＯＶＩＮＧ　セーブしておいた画像を平行移動させて画面上の画像と合成する。

(3) ＷＩＮＤＯＷ　画面上のカーソルで指定された範囲の画像とフロッピディデスク上の画像とを合成する。

以上の３通りの処理領域について，それぞれ以下のサブメニューに示す処理が行なえる。

(1) ＬＯＡＤ　ディスクに保存されている画像をそのまま画像メモリに書き込む。したがって，画面上の指定された範囲内の画像は画像Ｂでおき換えられる。

(2) ＡＮＤ　画像Ａと画像Ｂの対応する画素どうしで濃度値の各ビットの論理積をとる。

(3) ＯＲ　対応する画素どうしで濃度値の各ビットの論理和をとる。

(4) ＸＯＲ　対応する画素どうしで濃度値の各ビットの排他的論理和をとる。

(5) ＡＤＤ　対応する画素どうしの濃度値の和をその画素の濃度値とする。このとき，和が１５を超えた場合その画素の濃度値は１５におさえられる。

(6) ＡＶＥＲＡＧＥ　対応する画素どうしの濃度値の平均値をその画素の濃度値とする。平均をとったとき小数点以下は切り捨てられる。

(7) ＳＵＢ（Ａ－Ｂ）　画像Ａの画素の濃度値から対応する画像Ｂの濃度値を引いた結果を画像Ａの新らしい濃度値とする。結果が負になった場合その画素の濃度値は０におさえられる。

(8) ＳＵＢ（Ｂ－Ａ）　上とは反対に画像Ｂの濃度値から画像Ａの対応する画素の濃度値を引いた値を結果の濃度値とする。

(9) ＳＰＥＣＩＡＬ　ＬＯＡＤ　画像Ａの指定した濃度値をもった画素だけそれに対応する画像Ｂの画素データでおき換える。

３・６　ＨＩＳＴＧＲＡＭ(濃度値ヒストグラム)

ＲＩＧＩＯＮで設定した領域内に各濃度値をもった画素がいくつあるかを数え，その結果をディスプレイ上にヒストグラム表示する。

３・７　ＣＨＡＮＧＥ　ＶＡＬＵＥ（濃度変換）

指定領域の全画素の濃度を変換する。以下のサブメニューがある。

(1) ＲＥＶＥＲＳＥ（濃度値反転）

１５からもとの濃度値を引いたものを新らしい濃度値とする。

(2) ＴＨＲＥＳＨＯＬＤ（二値化）

しきい値以上の画素の濃度値を１５に，それ未満のものを０に変える。

(3) ＣＯＮＴＲＡＳＴ（コントラスト強調）

画像のコントラストが低いとき，コントラストを強調して画像を見やすくする。各画素のもつ濃度値ｖ（ａ≦ｖ≦ｂ）を次式によってｖ'（$v_1 \leqq v' \leqq vu$）に変換する。

$$v' = \frac{v_u - v_1}{b-a}(v-a) + v_1$$

小数点以下は切り捨てられる。

(4) ＰＩＣＫＵＰ　ＶＡＬＵＥ（特定濃度の抽出）

ある濃度をもった画素だけを取り出す。指定した濃度値をもつ画素を０に，残りの画素はすべて１５に変える。

(5) ＳＨＩＦＴ　ＶＡＬＵＥ

領域内の全画素の濃度値を指定した一定値だけ増加あるいは減少させる。入力する値は−15から15までで，結果が15以上のときは15に，０以下のときは０におさえられる。

(6) ＡＮＹ　ＣＨＡＮＧＥ(任意濃度への変換)

０から１５までの各濃度値をそれぞれに対応する指定値に変える。

３・８　ＣＯＰＹ（プリンタへの出力）

ディスプレイ上の画像のハードコピーをドットプリンタに打ち出す。また，各画素の濃度値を１６進

数の数字を用いてディスプレイ上の画素位置と同じ配置でプリンタに出力する。すなわち次のサブメニューがある。

(1) ＨＡＲＤ　ＣＯＰＹ　　１画素４＊４の１６点を用いて１６段階の濃度変化を表現する。

(2) ＶＡＬＵＥ　ＰＲＩＮＴ　　画像データを１画素１６進数の１文字に対応させて出力する。

３・９　ＭＡＳＫ（マスク処理）

３＊３個の画素のマスクを用いてフィルタ処理を行う。次の四つのサブメニューがある。

(1) ＡＶＥＲＡＧＥ（平滑化）

３＊３画素の濃度値の平均値をそのマスクの中央の画素の濃度値とする。ただし中央の画素の濃度の重みは周辺の画素の重みの２倍とする。平滑化により一点ノイズを除去できる。３＊３の画素の濃度値をｖ（ｘ，ｙ）とすると

$$(A)\quad v'(x,y)=\frac{1}{10}\sum_{i=-1}^{1}\sum_{j=-1}^{1}v(x+i,\ y+j)$$

(2) ＤＩＦＦＥＲＥＮＴＩＡＬ（境界検出）

$$v'(x,y)=\left|\sum_{j=-1}^{1}v(x+1,\ y+j)-\sum_{j=-1}^{1}v(x-1,\ y+j)\right|+\left|\sum_{i=-1}^{1}v(x+i,\ y+1)-\sum_{i=-1}^{1}v(x+i,\ y-1)\right|$$

すなわち，横方向の微分値の絶対値と縦方向の微分値の絶対値の和を微分値とする。これにより濃度境界の検出ができる。

(3) ＬＡＰＬＡＣＩＡＮ（境界検出）

$$v'(x,y)=9v(x,y)-\sum_{i=-1}^{1}\sum_{j=-1}^{1}v(x+i,\ y+j)$$

(4) ＡＮＹ　ＭＡＳＫ

３＊３のマスクに自分で決めることのできる重み $a_{i,j}$ をつけて処理を行う。

$$v'(x,y)=\mathrm{INT}\left(\sum_{i=-1}^{1}\sum_{j=-1}^{1}a_{i,j}v(x+i,\ y+j)\right)$$

３・１０　ＳＣＡＬＥ（画像の拡大・縮小）

画像を整数倍に拡大，あるいは整数分の一に縮小する。縦方向と横方向の倍率は別々の値をとることができる。

(1) ＥＮＬＡＲＧＥ（拡大）

一つの画像を縦・横ともに整数倍の画素数に拡大し，その全ての画素に対応するもとの画素の濃度値を入れる。

(2) ＲＥＤＵＣＥ（縮小）

縦横とも整数個の画素の平均濃度を求め，それを対応する縮小後の画素の濃度値とする。

３・１１　ＳＡＭＰＬＥ（画像の生成）

ＣＣＴＶカメラＮＥＭＣＯ　ＣＮ－１３０よりケイオー電子工業製のパーソナルコンピュータ用画像入力装置ＩＦＭ／ＰＣ－ＪＲに取りこんだ画像をＰＣ－９８０１Ｆ側に読み出し，フロッピディスクにセーブする。

4. プログラムの設計方針

パソコンによる画像処理は大型計算機に比べて処理速度とメモリ容量の点で劣っている。この弱点を補ない，かつ，パソコンの長所を生かしたソフトウエアを構成するために，以下の方針でプログラミングを行なう。

(1) プログラムの改造、追加が容易にできるようにするためにモジュール化を行ない，プログラムを見易くする。

初期設定，メインメニュー，共通サブルーチン，各処理ルーチンをそれぞれモジュール化する。

初期設定部では以下の作業が行なわれる。BASIC領域の上限の設定，変数の整数型宣言，BASICプログラムで使用する配列の宣言，FCBアドレスの機械語プログラム領域への書き込み，機械語プログラムのＬＯＡＤ。

共通サブルーチンには次のものがある。

ＴＩＴＬＥ　ＰＲＩＮＴ　メインメニューで選択した各処理のタイトルを表示する。

ＣＵＲＳＯＲ　ＤＲＡＷ　カーソルを表示する。

ＣＵＲＳＯＲ　ＥＲＡＳＥ　カーソルを消去する。

ＣＵＲＳＯＲ　ＤＡＴＡ　カーソルの座標データならびに処理領域のパラメータを求める。

ＷＲＩＴＥ　ＣＵＲＳＯＲ　ＤＡＴＡ　カーソルの座標データを機械語プログラム領域に書き込む。

ＷＲＩＴＥ　ＲＥＧＩＯＮ　ＤＡＴＡ　処理領域のパラメータを機械語プログラム領域に書き込む。

ＷＲＩＴＥ　ＶＡＬＵＥ　ＤＡＴＡ　濃度変換および画像合成で用いるテーブルデータｖ（　）を機械語プログラム領域に書き込む。

ＷＲＩＴＥ　ＰＩＣＴＵＲＥ　ＤＡＴＡ　カーソルデータおよび処理領域パラメータ，コメント文をファイルバッファに書き込む。

ＲＥＡＤ　ＰＩＣＴＵＲＥ　ＤＡＴＡ　ファイルバッファより上記データを読み出す。

ＲＥＡＤ　ＣＵＲＳＯＲ　ＤＡＴＡ　カーソルデータを機械語プログラム領域から読み出す。

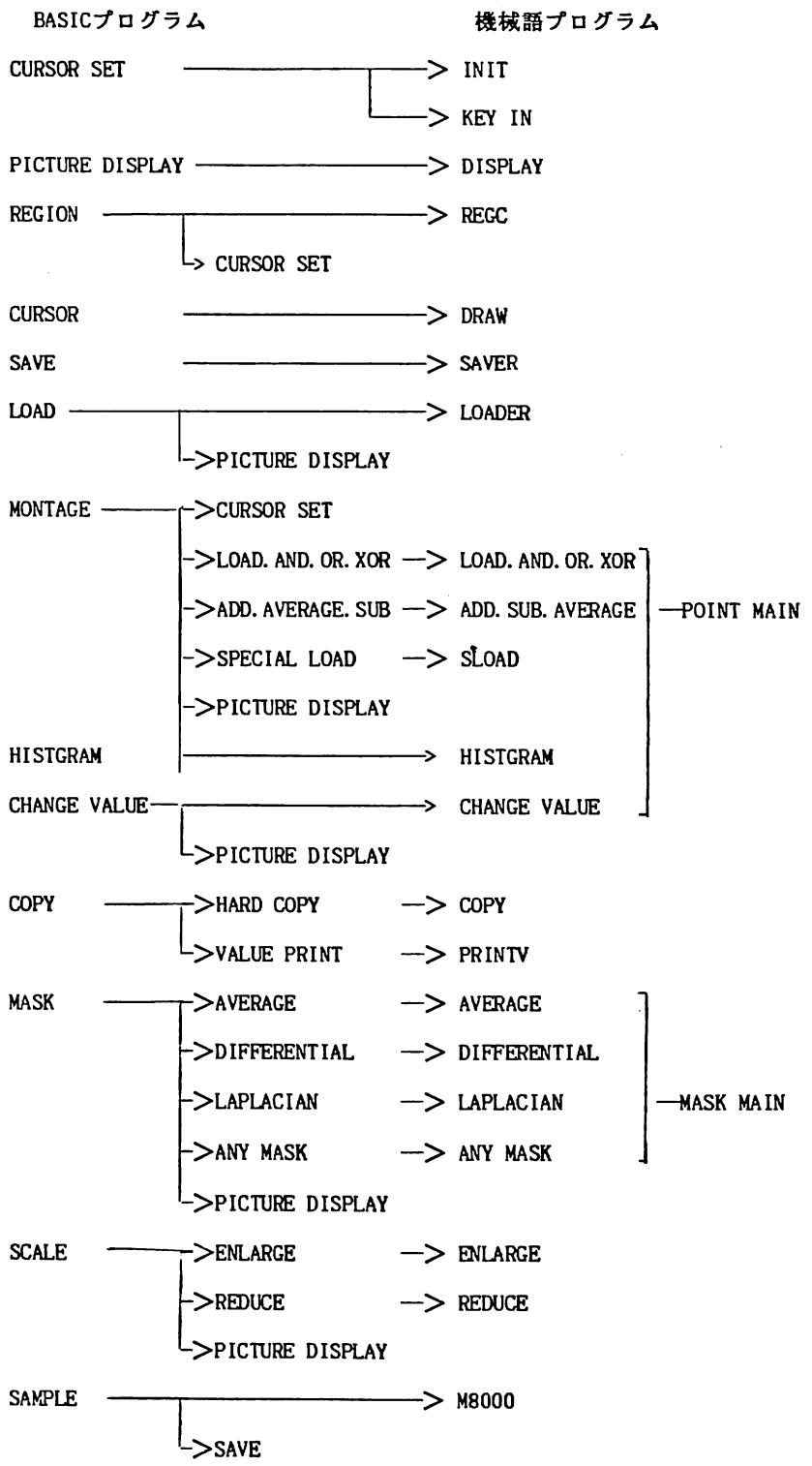

図　BASICプログラムと機械語プログラム

昭和 61 年 2 月

ＣＵＲＳＯＲ　ＳＥＴ　カーソルを動かして処理領域を設定する。
ＩＮＰＵＴ　ＦＩＬＥ　ＮＡＭＥ　オープンしたいファイルの名前を入力する。
ＩＮＰＵＴ　ＣＯＭＭＡＮＤ　№　メニュー選択したいコマンドの番号を入力する。
ＰＩＣＴＵＲＥ　ＤＩＳＰＬＡＹ　画像メモリ上の画像をディスプレイに表示する。

(2)　BASICプログラムと機械語プログラムからなるハイブリッド方式とする。

パソコンで一般的に使われている BASIC 言語はプログラミングは楽であるが，処理速度が遅く，大量のデータを処理しなければならない画像処理には適しない。会話型として計算を高速に行なわせるには機械語でプログラミングする必要があるが，この機械語によるプログラミングは簡単ではない。

プログラムを見易くし，その変更，追加を容易にするためと，処理の高速化をはかるためにはBASICと機械語の両方を用いてプログラミングするのが望ましい。BASICと機械語の使いわけは次のとおりである。

プログラム全体の流れをコントロールする部分(メニュー選択）や，初期設定，定数値設定など繰返し計算を行なわない部分はBASICでプログラミングする。

一方，メニューで選ばれた個々の処理部分や繰返し計算部分は機械語でプログラミングする。

計算に多くの時間を費やしているのは繰り返し計算を行っている部分であるが，この部分のプログラムの長さは他の部分の長さに比べて一般にそれほど長くはない。この繰返し部分だけを機械語でプログラミングすれば比較的楽にプログラミングでき，処理時間の節約効果は非常に大きい。

なお，機械語のプログラムを直接機械語命令で書くのは非常に困難である。実際にはＰＣ－９８０１Ｆの機械語モニタを利用してプログラミングを行なう。

(3)　整数計算を原則とする。

パソコンでの実数計算は多くの時間を必要とするが，整数計算なら高速にできて，メモリも少なくて済む。

実数データは１変数４バイトを必要とするのに対し，整数データなら１変数２バイトのメモリで済む。さらに，BASICでは１変数２バイトを要するが，機械語でプログラミングする場合，濃度レベルが４ビット１６段階なら平滑化やラプラシアンなどの途中計算や計算結果の値はたいていの場合８ビットに納まるので１変数１バイトで済む。

5.　プログラミングの手法

5・1　テーブルを用いた処理

たとえば濃度変換など，データｖに対して変換ｆを施してｖ'という結果を得る処理

$$v' = f(v)$$

は各画素ごとに施され，全画面の処理では６万回以上も繰返されなければならない。

ところが濃度値は高々１６通りしかなく，ｖ'もまた１６通りしかないので，変換前の濃度値 v_i （ｉ＝０，１，・・・，１５）を引数としてそれに対応する変換後の濃度値 v_i' を取り出すテーブルを作成しておくと，複雑な演算処理もテーブルからデータを読み出すだけで済み，大幅な計算時間の節約になる。

テーブルの活用はBASICと機械語の互いの欠点を補い，長所を生かしたプログラミングの手法であり，高速処理のための有効な手段である。

(1)　ＭＯＮＴＡＧＥのＡＤＤ．ＡＶＥＲＡＧＥ．ＳＵＢならびにＣＨＡＮＧＥ　ＶＡＬＵＥで濃度変換に用いるテーブルの作成はBASICプログラムで行なう。

(2)　ＭＡＳＫのＡＶＥＲＡＧＥ（平滑化）では各画素の濃度値 V_{ij} は０から１５の値なので変換式(A)における総和のとり得る値は０から１５０の整数値であり，これを１０で割った値を求めるテーブルを用意しておけば平均値を求めるのに機械語プログラムの中で総和の計算だけを行えばよい。

(3)　任意マスクの処理プログラムでは３＊３のマスクに対応して九つのテーブルを用意し，濃度値を引数として濃度値＊重みデータの積 $a_{ij} * v_{ij}$ を各テーブルから取り出し，その和を計算する。

濃度値＊重みデータのテーブルは機械語プログラムで作成する。

(4)　ＳＣＡＬＥのＲＥＤＵＣＥでは平均値テーブルを使用するが，これは，平均すべき全画素の濃度値の和をその画素数（縦横の各縮小率の積）Ｋで割ればよい。各画素のとり得る濃度値は０から１５の整数値だから，Ｋ個の画素の濃度値の和のとり得る値は０から１５Ｋである。したがって１５Ｋ＋１個のテーブルを作ればよい。これは機械語プログラムで行なう。

5・2　機械語プログラミングの手法

(1)　BASICプログラムから機械語プログラムを呼

び出すにはCALL文を用いる。その際データの引渡しはPOKE文で行う。

各処理ルーチンで機械語プログラム領域に引渡されるデータはそれぞれ次のとおりである。

ＲＥＧＩＯＮではWRITE CURSOR AND REGION DATAでデータを引渡す。

ＳＡＶＥではWRITE PICTURE DATAでデータをファイルバッファに書きこむ。

ＬＯＡＤではWRITE REION DATAでデータを引渡す。

MONTAGE では WRITE REGION DATA ならびに WRITE CURSOR DATAを実行しウインドウ処理フラグと WINDOW使用のときはさらにそのパラメータを，また，コマンド LOAD. AND. ORのときはオペレーションコードが，コマンドADD. AVERAGE. SUBのときはオペレーションコードと濃度値テーブルｖ(　) が，SPECIAL LOADのときは指定濃度値が機械語領域に書き込まれる。

ＣＨＡＮＧＥ　ＶＡＬＵＥでは濃度値テーブルｖ(　) が引き渡される。

ＨＡＲＤ　ＣＯＰＹでは印字モードを８ビットドット対応グラフィックモードに切り換えるためにプリンタに送るべき６バイトの情報をPOKEする。

ANY MASKではマスクの重みデータが引き渡される。

ＳＣＡＬＥでは縦、横それぞれの拡大（縮小）率が引渡され，WRITE PICTURE DATA が CALLされる。

(2)　プログラムによるプログラムの書き換え

機械語プログラムでは演算コードやジャンプ先（コール先）の番地を変えるなど，プログラム自身を書き換える手法を用いる。これはプログラムを有効に使いメモリを節約するための手法である。

書き換えが行われる部分は次のとおりである。

ＭＯＮＴＡＧＥではLOADER のデータセグメントの書き換え

ＨＩＳＴＧＲＡＭ，ＣＨＡＮＧＥ　ＶＡＬＵＥ，ＡＮＤ．ＯＲ．ＸＯＲ，ＡＤＤ．ＳＵＢでそれぞれ　ＰＯＩＮＴ　ＭＡＩＮよりのコール先アドレス

AND. OR. XOR ならびに ADD. SUBではさらにオペレーションコードならびにWIND処理後のジャンプ先アドレスの書き換え

MASK処理ではMASK　MAINよりAVERAGE，DIFFERNTIAL，LAPLACIAN，ANY MASK 各処理のコール先アドレス

SCALE の ENLARGE, REDUCE ではそれぞれデータセグメントベースの書き換え

ＳＡＭＰＬＥではＳＡＶＥＲの書き換え(データセグメントベース）が行なわれる。

(3)　割り込みの利用

周辺装置との入出力操作を行う部分のプログラミングにはＰＣ－９８０１Ｆシステムの割り込みを利用する。

各処理ルーチンで用いられる割り込みは以下のとおりである。

ＤＩＳＰＬＡＹではＡＨ＝４５としてＩＮＴ１８（ＣＲＴ　ＢＩＯ）

ＬＯＲＤＥＲではＡＨ＝６としてＩＮＴＣ６（＊ＧＥＴ）

ＳＡＶＥＲではＡＨ＝７としてＩＮＴＣ６（＊ＰＵＴ）

ＣＯＰＹ，ＰＲＩＮＴＶではＡＨ＝１１としてＩＮＴ１Ａ（ＰＲＩＮＴＥＲ　ＢＩＯ）

ＫＥＹＩＮではＡＨ＝０としてＩＮＴ１８（ＫＢＢＩＯ）

＊ＧＥＴ，＊ＰＵＴへ分岐するには，ＤＩレジスタに０Ａ，ＢＸレジスタにＦＣＢアドレス，ＤＳならびにＥＳレジスタに００６０をそれぞれ格納する。

また，ＣＲＴ　ＢＩＯのグラフィックスクリーンへのドット列の書き込みへ分岐するには，ＤＳレジスタへ引数ワークエリアのセグメントベース0060を、ＢＸレジスタへ引き数ワークエリアのオフセットアドレス０６４０を，ＣＨレジスタにＢ０を，ＥＳレジスタに濃度値データのセグメントベースを，〔ＢＸ〕に濃度値を，〔ＢＸ＋８～９〕に画素のスクリーンｘ座標を，〔ＢＸ＋０Ａ～０Ｂ〕にｙ座標を，〔ＢＸ＋０Ｃ～０Ｄ〕にドット列の長さをそれぞれ格納して分岐する。

ＡＨ＝１１，ＩＮＴ　１Ａのとき　レジスタＡＬに出力すべきデータをいれて分岐する。

ＡＨ＝０，ＩＮＴ　１８のとき　レジスタＡＸにキーボードからの入力データを格納してもどる。

6.　処理の実行時間

１６ビットＣＰＵ８０８６の機械語によるプログラミングの効果をみるために，各処理の実行時間を計測した。以下に示す数値は全領域（２５６＊２５６画素）を処理する場合の時間である。ただし，ＳＣＡＬＥのＥＮＬＡＲＧＥに限り原画像が１００＊１００画素の場合の数値である。

ＤＩＳＰＬＡＹ　　３３秒

SAVE	87秒
LOAD	8秒
MONTAGE	
AND, OR, XOR	3秒
ADD, SUB	4秒
HISTGRAM	4秒
CHANGE VALUE	
REVERSE	1秒
HARD COPY	288秒
MASK	
AVERAGE	6秒
DIFFERNTIAL	8秒
LAPLACIAN	7秒
ANY MASK	18秒
SCALE	
ENLARGE	3秒
REDUCE	5秒

7. あとがき

このプログラムは参考文献1のPC−8001用のプログラムをPC−9801F用に移植したものである。原プログラムからは種々のプログラムテクニックを学ばせていただいた。記して謝意を表わす。

参考文献

1. 森本　吉春　画像処理　培風館　1984年
2. 田村　秀行　コンピュータ画像処理入門　総研出版　1985年
3. 岡村秀一郎　PC−9801　キーボード・CRT／モニタ　解析マニュアル　秀和システムトレーディング　1984年
4. 井上　智博　PC−9801グラフイクス　解析マニュアル　秀和システムトレーディング　1984年
5. 山内　直　PC−9801　ディスクシステム　解析マニュアル　秀和システムトレーディング　1985年