数理解析研究所講究録
第455巻 1982年 191-206

多値論理を用いた画像処理専用プロセッサの構成

東北大学工学部　亀山充隆
樋口龍雄

1. まえがき　論理演算を用いて基本的な画像処理を行うために種々の提案がなされている[1]~[3]。しかしながら，これらはほとんど2値論理に基づくハードウェアを前提としたものであり，濃淡画像やいくつかの色からなるカラー画像などの処理には，さらに効率のよい処理技術が望まれる。本稿では，濃淡画像などにおいて，画素情報が直接多値に対応していることに着目し，多値論理を用いた新しい処理方式に基づく多値画像処理専用プロセッサを提案する。

ところで，多値論理は，1977年，米国半導体メーカにより多値集積回路ファミリが試作されて以来[4]，急速にその具体的利点が明らかにされている。最近では，4値ROMが実用化されるなど，今後，ますますその発展が期待される[5]。多値論理を用いた画像処理の考えは，その概念が文献(6)に報告されているが，具体的な利点や構成法などについては，十分

明らかにされていないのが現状である。

本稿では，このような背景のもとに，多値論理を用いたアルゴリズムとともに，多値論理によるハードウェアを前提とした画像処理専用アレイプロセッサの構成法について述べている。本アレイプロセッサは，BASEで代表されるようなBAP(Binary Array Processor)[7]を多値へ拡張したもので，MVAP(Multiple-Valued Array Processor)と呼ぶことにする。MVAPの最大の特長は，演算部が2値論理と比較して豊富な論理機能をもち，処理にも多様性が存在すると考えられる。このため，多値論理を用いた系統的なアルゴリズムすなわち論理関数の決定法が重要となってくる。このような多値論理を用いた系統的かつ効率的な画像処理アルゴリズムについて検討が加えられている。

2.画像処理用多値アレイプロセッサ　以下，標本化多値画像を考察することにする。このような画像演算子空間の代数的記述およびその性質は文献(8)(9)などにおいて述べられているが，ここでは，MVAPによる多値画像処理に必要な諸定義を上げる。

〔定義1〕(r値画像)　すべての整数の集合をI，I値からなる真理値の集合を$L=\{0,1,\cdots,r-1\}$とする。$I\times I$からLへの写像をr値画像と呼び，画像要素(i,j)の点に

おける値をその濃度値 $f_{ij} \in L$ とする。これにより得られた画像を $F=\{f_{ij}\}$ とする。■

ここで，濃度値 $f_{ij} \in L$ と，実際の画像における濃淡レベル，色などとは，あらかじめ対応関係を定めておくものとする。すなわち，各画素情報は L に対応させたまま種々の処理を行うことを考える。

〔定義2〕（点演算子） 画像 $F=\{f_{ij}\}$ から，画像 $G=\{g_{ij}\}$ を求める演算子 ψ である。$g_{ij}=\psi(f_{ij})$ for $\forall(i,j) \in I \times I$．但し，$\psi$ は，任意の T 値1変数関数である。■

〔定義3〕（2項点演算子） 2つの画像 $F=\{f_{ij}\}$, $G=\{g_{ij}\}$ から画像 $H=\{h_{ij}\}$ を求める演算子 Φ である。$h_{ij}=\phi(f_{ij}, g_{ij})$ for $\forall(i,j) \in I \times I$．但し，$\phi$ は，任意の T 値2変数関数である。■

MTAPにおいては，各サブプロセッサは，図1のような構造であり，これらが各画素にアレイ状に配置されるとともに，マイクロプログラムにより制御される。近傍画素入力として，図2に示される8近傍を考える。マイクロプログラムの近傍画素選択フィールド q により選択された $N_1 \sim N_8$ のうちのいくつかに対して，式(1)のように近傍関数演算が行われる。

$$P=\bigvee_{i=1}^{8}(q_i \wedge N_i) \qquad (1)$$

但し，$\vee$ と $\wedge$ は，それぞれ $OR(x_1,x_2)=max(x_1,x_2)$，

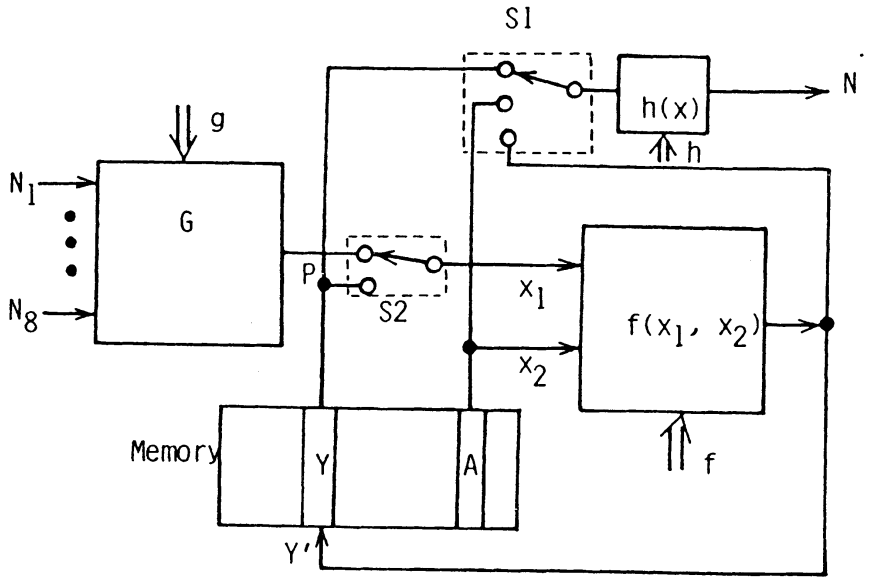

図1 MVAPのサブプロセッサ

N_1	N_2	N_3
N_8		N_4
N_7	N_6	N_5

図2 近傍画素の定義

$AND(x_1, x_2) = min(x_1, x_2)$である。また，$q_i$は，$N_i$が選択されるときのみ$r-1$の値をとり，その他は0である。多値論理演算回路$f(x_1, x_2)$においては，任意のr値2変数関数をマイクロプログラムの関数選択フィールドfにより決定することができる。このため，任意の論理関数を容易に実現できる多値Tゲート[10][11]を基本論理ブロックとして使用する。多値Tゲートはユニバーサルロジックモジュールの1種であり，次のように定義される。

$$T(p_0, p_1, \cdots, p_{r-1}; x) = p_i, \quad \text{if } x = i \tag{2}$$

任意の論理関数は，式(3)のように標準展開される。

$$f(x_1, x_2, \cdots, x_n) = T(f(x_1, x_2, \cdots, x_i=0, \cdots, x_n), f(x_1, x_2, \cdots, x_i=1, \cdots, x_n), \cdots, f(x_1, x_2, \cdots, x_i=r-1, \cdots, x_n); x_i) \quad (3)$$

多値Tゲートはハードウェア実現も容易であり，種々の多値論理演算や近傍演算を効率的に行うことができる。また，S1からの出力は，多値1変数関数$h(x)$を施すことにより，他のモジュールの近傍画素入力として使用される。

3.多値論理演算　図3に多値論理演算を行うためのMTAPの構造が示されている。r値2変数関数の種類はr^{r^2}個であり，例えば，2値と3値では，それぞれ16，19638種類となる。従って，多値論理においては，非常に豊富な論理構造

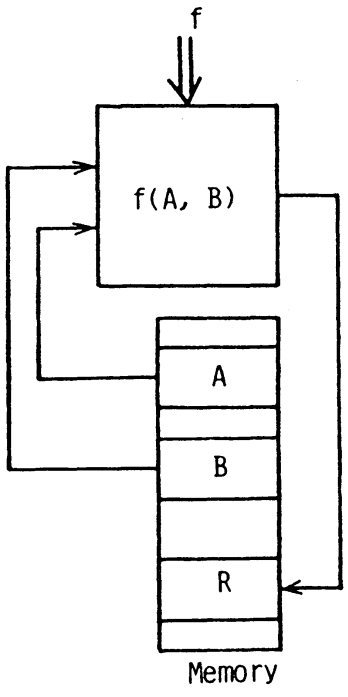

表1

Threshold-2

x	f(x)
0	0
1	0
2	3
3	3

Inverting

x	f(x)
0	3
1	2
2	1
3	0

Cycling

x	f(x)
0	1
1	2
2	3
3	0

Difference

x_1 \ x_2	0	1	2	3
0	0	1	2	3
1	1	0	1	2
2	2	1	0	1
3	3	2	1	0

Mean

x_1 \ x_2	0	1	2	3
0	0	1	1	2
1	1	1	2	2
2	1	2	2	3
3	2	2	3	3

図3　多値論理演算を行うための構造

を利用することにより，多様性のある画像処理を行うことができる。点演算では，しきい値処理，反転，巡回などを，2項点演算では，MAX，MIN演算，マスキング，論理差，平均などの意味をもった処理を行うことができる。これらの4値における関数の例を表1に示しておく。

4. 非再帰的近傍演算　本演算は，図4に対応して式(4)のように記述することができる。

$$R \leftarrow f(A,P) \text{ where } P \leftarrow NN(\langle list \rangle) \text{ of } A \text{ Edge} \langle value \rangle \quad (4)$$

但し，fは，任意の7値2変数関数を，$\langle list \rangle \subseteq \{1,2,\cdots,7,8\}$ はどの近傍が選択されているかを，また，$\langle value \rangle$ は端の部分の近傍画素値を示している。

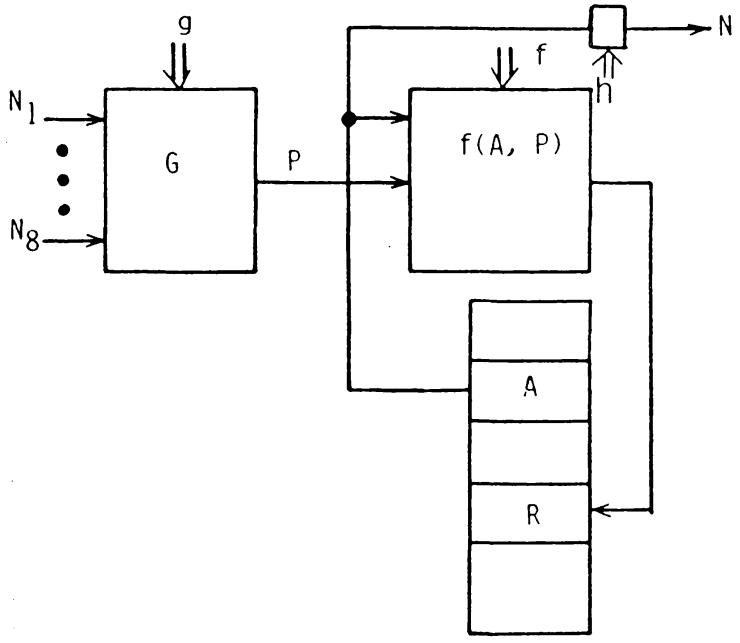

図4　非再帰的近傍演算の構造

非再帰的近傍演算は，基本的に入力画像に対するテンプレートマッチングに帰着させることができ，これは，一般的に式(5)のような状態遷移で記述できる。(12)

$$\delta:\ \begin{bmatrix} \overline{v}_1' & \overline{v}_2' & \overline{v}_3' \\ \overline{v}_8' & \overline{v}_0' & \overline{v}_4' \\ \overline{v}_7' & \overline{v}_6' & \overline{v}_5' \end{bmatrix} \rightarrow \overline{V}^1, \cdots, \begin{bmatrix} \overline{v}_1^{\mu} & \overline{v}_2^{\mu} & \overline{v}_3^{\mu} \\ \overline{v}_8^{\mu} & \overline{v}_0^{\mu} & \overline{v}_4^{\mu} \\ \overline{v}_7^{\mu} & \overline{v}_6^{\mu} & \overline{v}_5^{\mu} \end{bmatrix} \rightarrow \overline{V}^{\mu}, \cdots, \begin{bmatrix} \overline{v}_1^{k} & \overline{v}_2^{k} & \overline{v}_3^{k} \\ \overline{v}_8^{k} & \overline{v}_0^{k} & \overline{v}_4^{k} \\ \overline{v}_7^{k} & \overline{v}_6^{k} & \overline{v}_5^{k} \end{bmatrix} \rightarrow \overline{V}^{k} \quad (5)$$

今，式(5)で示されるテンプレートの1つ $(\overline{v}_0^{\mu}, \overline{v}_1^{\mu}, \cdots, \overline{v}_8^{\mu}) \rightarrow \overline{V}^{\mu}$ を考える。Pが近傍画素入力の論理和となっているため，1値の論理和項(Sum Terms)の導出を行えばよい。$\overline{v}_i^{\mu} = c$ $(c \in \mathcal{L})$ となる近傍リストの集合を $\{I_c\}$ とする。従って，入力画像Aに対する $\{I_c\}$ の近傍に関するマッチングは，$R(Initial)=0$ として次のように求めることができる。

$$R \leftarrow OR(P, R) \text{ where } P \leftarrow NN(\langle I_c \rangle) \text{ of } \overline{A^{cc}} \qquad (6)$$

但し，$\overline{x^{cc}}$ はコンプリメントリテラルであり，$x=c$ のとき0，その他のとき $r-1$ の値をとる。式(6)を $c=0$ から $r-1$ まで実行することにより，$R=0$ のままであれば，近傍画素は全て一致している。従って，最後にテンプレート演算結果は式(7)となる。

$$R \leftarrow f(R, A) \qquad (7)$$

但し，$f(0, \overline{v}_0^{\mu}) = \overline{V}^{\mu}$，その他のとき $f(R, A) = A$ である。

例として，4値論理におけるパターンマッチングを考える。

$$\begin{bmatrix} 1 & 1 & 1 \\ 3 & 0 & 3 \\ 3 & 2 & 1 \end{bmatrix} \rightarrow 1 \qquad \begin{bmatrix} d & d & d \\ d & d & d \\ d & d & d \end{bmatrix} \rightarrow 0 \tag{8}$$

表2のように，$\mathcal{L}=\{0,1,2,3\}$ と画像の記号との対応関係を定めておく。式(8)のようなテンプレートマッチングは，以下のようなアルゴリズムとなる。

```
R(Initial) = 0
R ← OR(P, R) where P ← NN(1, 2, 3, 5) of T(3, 0, 3, 3 ; A)
R ← OR(P, R) where P ← NN(6) of T(3, 3, 0, 3 ; A)                (9)
R ← OR(P, R) where P ← NN(4, 7, 8) of T(3, 3, 3, 0 ; A)
R ← T(T(1, 0, 0, 0 ; R), 0, 0, 0 ; A).
```

この処理例を図5に示す。

実際の画像処理においては，上述の手順に基づき，さらにアルゴリズムの簡単化が行える場合が多い。以下，具体例を述べる。

(α) 雑音除去　ここでの雑音除去は，孤立点を除去することである。このため，式(10)のようなテンプレートを考える。

表2

0	1	2	3
	o	·	×

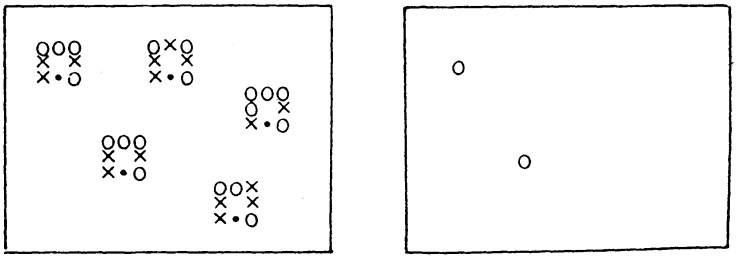

図5 テンプレート演算の例

但し，dは don't care を示している。

$$\begin{bmatrix}0&0&0\\0&d&0\\0&0&0\end{bmatrix}\to 0 \quad \begin{bmatrix}1&1&1\\1&d&1\\1&1&1\end{bmatrix}\to 1 \quad \begin{bmatrix}2&2&2\\2&d&2\\2&2&2\end{bmatrix}\to 2 \quad \begin{bmatrix}3&3&3\\3&d&3\\3&3&3\end{bmatrix}\to 3 \qquad (10)$$

アルゴリズムは，以下のように表すことができる。

```
R ← T(0, d, d, A ; P) where P ← NN(all 8) of T(0, 3, 3, 3 ; A)
R ← T(1, d, d, R ; P) where P ← NN(all 8) of T(3, 0, 3, 3 ; R)
R ← T(2, d, d, R ; P) where P ← NN(all 8) of T(3, 3, 0, 3 ; R)
R ← T(3, d, d, R ; P) where P ← NN(all 8) of T(3, 3, 3, 0 ; R)
```
(11)

さらに，これにより得られた画像Rに対しても，ブランク(0)と他の同一の記号(1, 2, 3)とで囲まれたような点も雑音とみなすと，次のようなアルゴリズムにより，この雑音を除去できる。

```
R←T(T(0, 1, 1, 1 ; R), d, d, R ; P) where P←NN(all 8) of T(0, 0, 3, 3 : R)
R← T(T(0, 2, 2, 2 ; R), d, d, R ; P) where P←NN(all 8) of T(0, 3, 0, 3 ; R)
R←T(T(0, 3, 3, 3 ; R), d, d, R ; P) where P←NN(all 8) of T(0, 3, 3, 0 ; R)
```
(12)

入力画像Aを図6とすると，図7が得られる。

(b) 論理微分　横方向の論理微分は，次のようなテンプレート演算である。但し，$a, b \in L$ である。

$$\begin{bmatrix}d&d&d\\a&b&d\\d&d&d\end{bmatrix}\to (b-a) \bmod 4 \qquad (13)$$

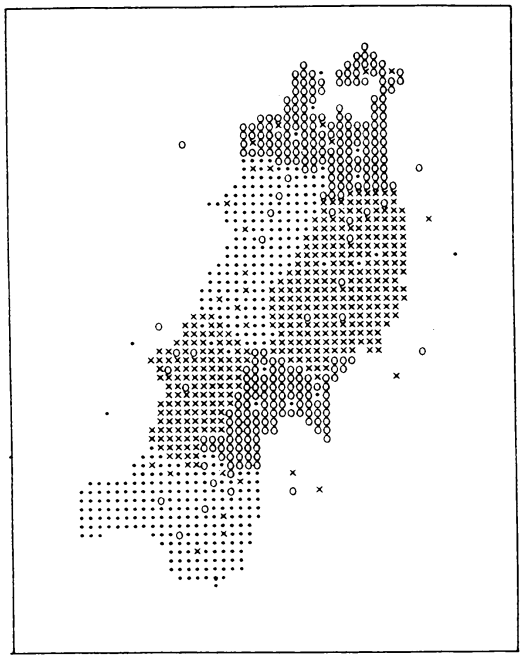

図6 入力画像

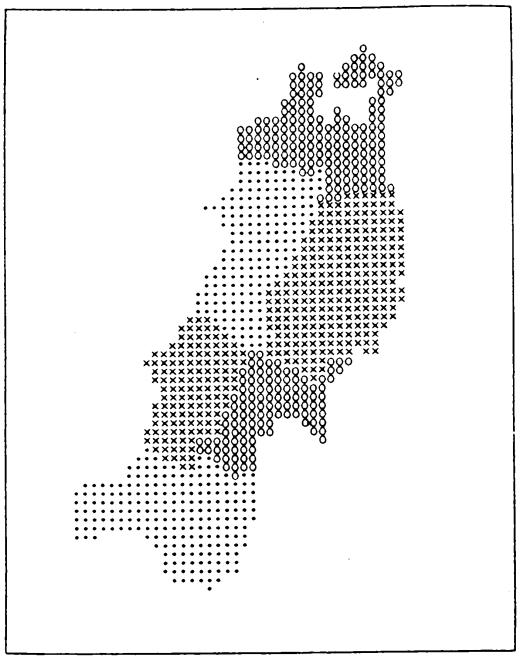

図7 雑音除去

従って，アルゴリズムは以下のようになる。

```
R←T(P, T(1, 2, 3, 0 ; P), T(2, 3, 0, 1 ; P), T(3, 0, 1, 2 ; P) ; A)
    where P←NN(8) of T(0, 3, 2, 1 ; A).
```

(14)

縦方向の論理微分も同様にして次のようなアルゴリズムとなる。

```
R←T(P, T(1, 2, 3, 0 ; P), T(2, 3, 0, 1 ; P), T(3, 0, 1, 2 ; P) ; A)
    where P←NN(2) of T(0, 3, 2, 1 ; A).
```

(15)

これらの処理例を，図8，9に示す。

(C) ハッチング処理　ハッチング処理は，疎に拡大された画像から，種々の模様を作ることができる。図10を入力画像として，×を右下に，○を下方向にハッチングするアルゴリズムを以下に示す。

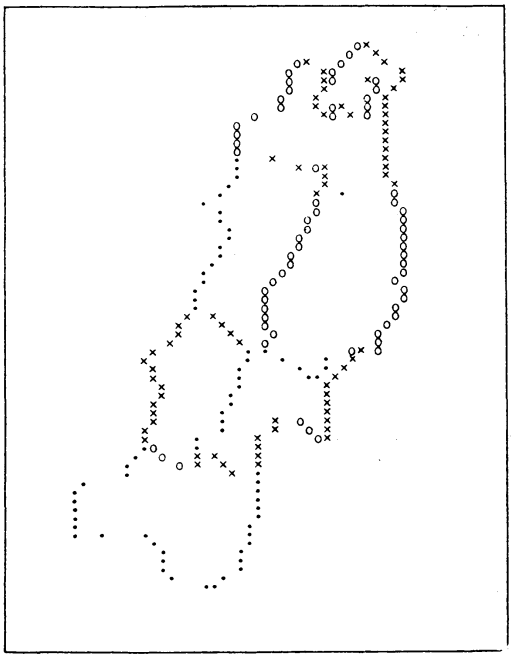

図8 横方向微分

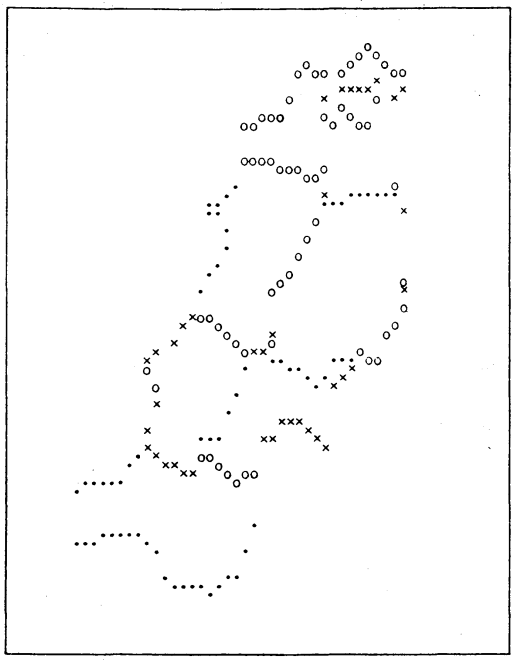

図9 縦方向微分

R←T(A, d, d, 3 ; P) where P←NN(1) of T(0, 0, 0, 3 ; A)

R←T(A, d, d, 1 ; P) where P←NN(2) of T(0, 3, 0, 0 ; A).

(16)

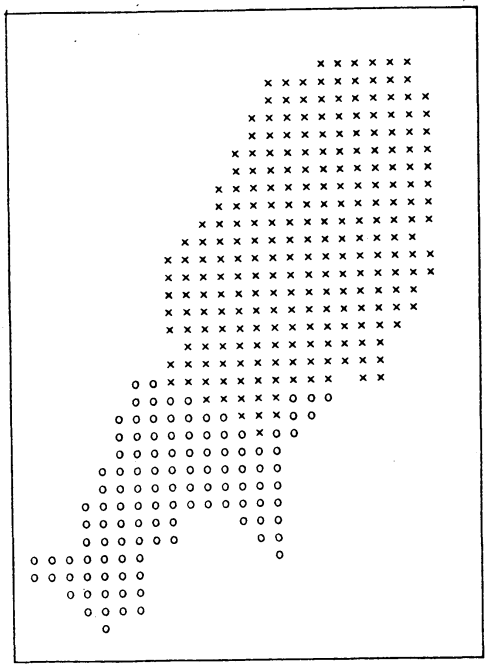

図10 拡大画像

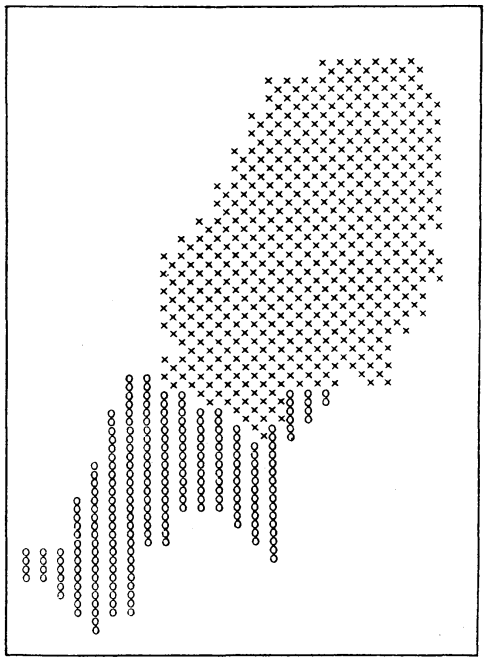

図11 ハッチング処理

(d) 輪郭線の抽出　輪郭線の抽出は，式(17)のテンプレート演算である。

$$\begin{array}{|ccc|}\hline a & a & a \\ a & d & a \\ a & a & a \\ \hline\end{array} \rightarrow 0 \qquad (17)$$

```
R←T(0, d, d, A ; P) where P←NN(all 8) of T(3, 0, 3, 3 ; A)
R←T(0, d, d, R ; P) where P←NN(all 8) of T(3, 3, 0, 3 ; R)      (18)
R←T(0, d, d, R ; P) where P←NN(all 8) of T(3, 3, 3, 0 ; R).
```

図7を入力画像として，図12が得られる。

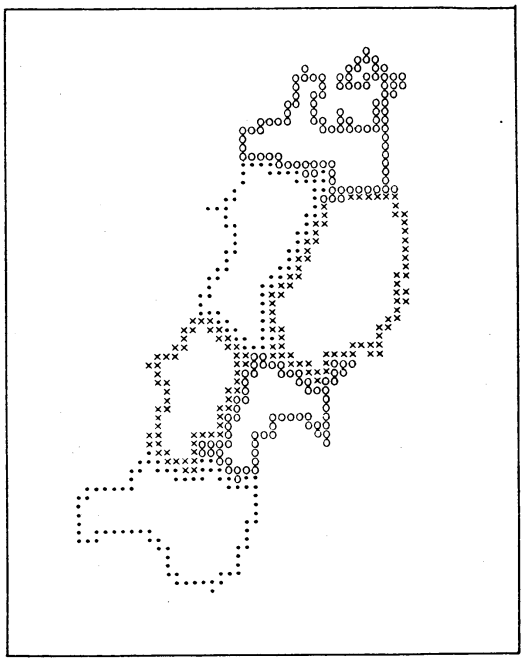

図12 輪郭線の抽出

5. 再帰的近傍演算　再帰的近傍演算は，図13のようなMTAPの構造に対応して，式(19)のように記述される。

```
Repeat R← f(A,P) where P←NN(<list>)
of R Edge<value>                                    (19)
```

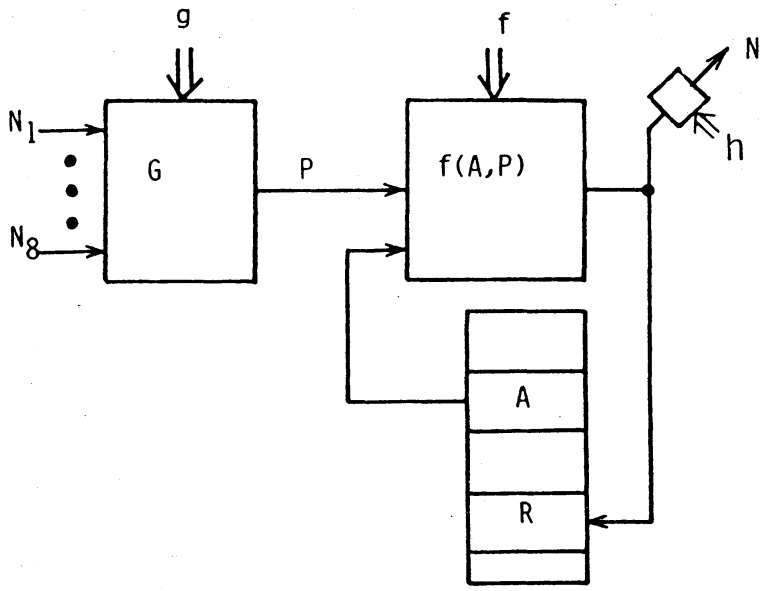

図13 再帰的近傍演算の構造

すなわち，Rの値を逐次使用して，Rの値が変化しなくなるまで演算をくり返す。本演算においても，テンプレート演算を再帰的にくり返すことにより，種々の処理を行うことができる。

(a) 連結成分の抽出　8近傍に連結している成分を抽出するには，次のようなアルゴリズムを行えばよい。

```
R1(Initial) ; given

Repeat R1←T(0, d, d, T(0, 1, 0, 0 ; A) ; P) where P←NN(all 8) of
          T(0, 3, 0, 0 ; R1)

R2(Initial) ; given

Repeat R2←T( 0, d, d, T(0, 0, 2, 0 ; A) ; P) where P←NN(all 8) of
          T(0, 0, 3, 0 ; R2)
R3(Initial) ; given

Repeat R3←T( 0, d, d, T(0, 0, 0, 3 ; A) ; P) where P←NN(all 8) of
          T(0, 0, 0, 3 ; R3)

R←OR(R1, R2, R3).
```

(20)

図15は，図7を入力画像，Rの初期状態(OR(R1, R2, R3))を図14として連結成分を抽出したものである。

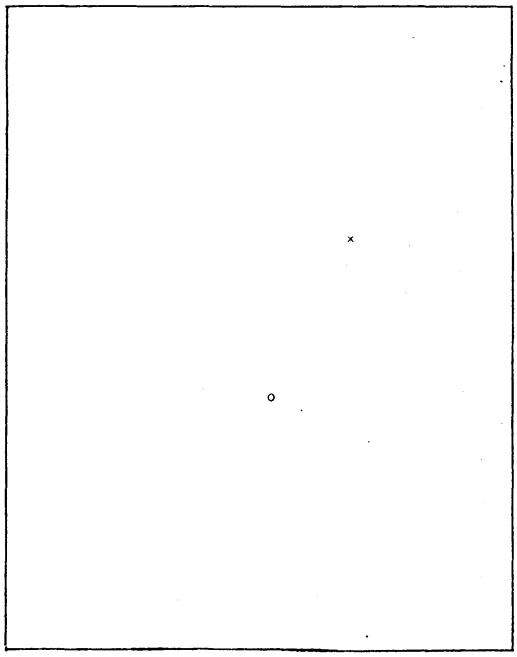

図14 初期状態

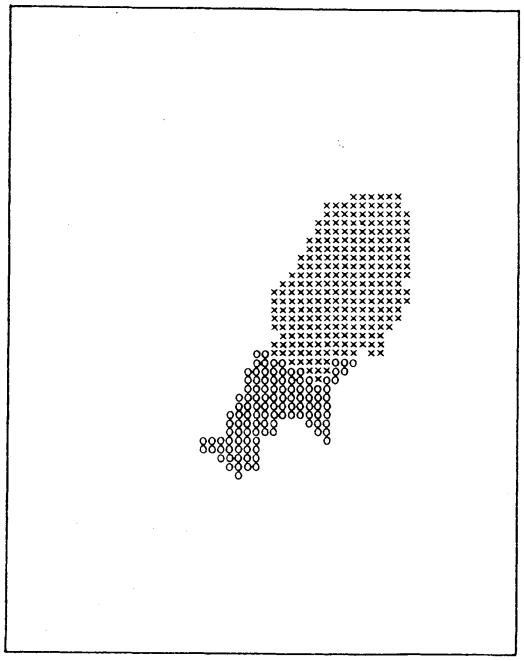

図15 連結成分の抽出

(ヒ) 幅の検出 縦幅及び横幅の検出は次のようになる。

```
Repeat R←T(A, 1, 2, 3 ; P) where P←NN(8) of R                    (21)
```

```
Repeat R←T(A, 1, 2, 3 ; P) where P←NN(2) of R                    (22)
```

(C) 拡大 拡大は，単なるデータのシフトであるが，ここでは，2倍に拡大するアルゴリズムを示す。Sは，拡大する始点を示す。まず，右方向への拡大は，式(23)となる。

```
R(Initial) = 0, S(Initial) ; given
Repeat R←OR(R, T(0, A, A, A ; S))
       A←NN(8) of A                                              (23)
       S←NN(8) of S
       S←NN(8) of S.
```

次に，これを下方へ拡大する。

```
R(Initial) = 0, S(Initial) ; given
Repeat R←OR(R, T(0, B, B, B ; S))
       B←NN(2) of B
       S←NN(2) of S
       S←NN(2) of S.
```
(24)

入力画像を図15とすると，図10の拡大画像が得られる。

(d) 論理積分 論理微分の逆操作である。横方向の積分は以下のようになる。

```
R(Initial) = 0
Repeat R←T(A, T(1, 2, 3, 0 ; A), T(2, 3, 0, 1 ; A), T(3, 0, 1, 2 ; A) ; R)
       A←NN(8) of A.
```
(25)

(e) 穴うめ 周囲が閉じている連結成分で囲まれた内部をうめる操作であり，輪郭線の抽出の逆操作である。

```
R(Initial) = 0
Repeat R←T(0, d, d, T(3, 0, 3, 3 ; A) ; P) where P←NN(all 8) of
         T(0, 0, 0, 3 ; R) Edge 3.
R←OR(A, T(1, 0, 0, 0 ; R)).
```
(26)

上述の手順は，1で囲まれた部分の穴うめを示している。

6. むすび 多値論理を用いることにより，いくつかのレベルをもつ濃淡画像やいくつかの色，記号などで表現される画像をデコードすることなしに多値に1対1に対応させたまま，系統的に処理できることを明らかとした。今後の問題として，

MVAPのLSI化を前提とした多値ハードウェアの評価，また，多値論理の豊富さをさらに生かしたアルゴリズムの系統的記述と簡単化などが考えられる。

文献

(1) A. Rosenfield: Picture Processing by Computer, Academic Press, New York (1969).

(2) 島脇，横井：画像処理のアルゴリズム，情報処理，vol.21, 6, p.613 (昭55).

(3) 木戸出：画像処理用ハードウェア，情報処理，vol.21, 6, p.620. (昭55).

(4) T. T. Dao, E. J. McCluskey and L. K. Russell: Multivalued Integrated Injection Logic, IEEE Trans. Comput., vol. C-26, p.1233 (1977).

(5) M. Stark: Two Bits per Cell ROM, Proc. of COMPCON-Spring. IEEE 81-CH1626-1 (1981).

(6) D.C. Rine: Picture Processing Using Multiple-Valued Logic, Proc. of the 1981 ISMVL, p.73 (1981).

(7) A.P. Reeves: A Systematically Designed Binary Array Processor, IEEE Trans. Comput., vol. C-29, p.278 (1980).

(8) 横井，島脇，福村：標本化図形の演算系の代数的構造とその応用，電子通信学会論文誌，60-D, 6, p.430 (1977).

(9) 安居院，山之内，中嶋：ディジタル画像の代数的記述とその応用，信学技報，PRL 80-109, p.99 (昭55).

(10) 亀山，樋口：ユニバーサルロジックモジュールに基づく多値論理回路網の合成，信学論(D), J60-D, 5, p.355 (昭52).

(11) T. Higuchi and M. Kameyama: Static-Hazard-Free T-Gate for Ternary Memory Element and Its Application to Ternary Counters, IEEE Trans. Comput., vol. C-26, p.1212 (1977).

(12) B. Kruse: A Parallel Picture Processing Machine, IEEE Trans. Comput., vol. C-22, p.1075 (1973).