

# 領域情報の専用処理システム

中 野 旭

## A Processing System for Region Information

By

Akira NAKANO

**Abstract:** Some investigation on fundamental procedures for handling region information are performed, then a method and a device for region information processing are described in this paper. A model and two sets of graphic display devices which have utilized that philosophy of region information processing are designed and manufactured, some details about those devices are described in the latter part of this paper.

### 概 要

領域情報処理を情報処理分野の中でも独自性の強い分野として考え、従来の処理手法に拘束されないでその処理手順を吟味した。本論文にはそこで得られた基礎手順と、これをその手順通りに実行する専用処理システムの構想が述べられている。更にこの構想を基盤とするグラフィックディスプレイ装置を製作したので、これについての概要が述べられている。

### 1. ま え が き

数値を対象とした情報処理には、算術的色彩の濃い処理と幾何的色彩の濃い処理と、二つの性格の異なるグループが存在している。現時点ではこれらのほとんどについての処理がいわゆるノイマン型の汎用電子計算機システムによって実行されているが、四則演算をその主たる基本機能としてまとめられたこれら計算機にとって後者の処理は決して能率の良いものではなく、これが領域に関する問題等の処理に対して専用のハードウェアシステムを求める気運につながっているであろう。

宇宙等特殊環境下に物体を到達させるためのシステムがこの領域に関する情報の処理と深い関係を持つであろうことには疑う余地も無いが、特に到達体に搭載されるシステムではその機能、重量、容積、信頼性等に課せられる条件は極めて厳しいものであろう。この観点からも領域情報の処理の問題を再吟味し、そこで見出された基本手順を基本機能とする専用システムの実現を図るべきものと考えている。たまたま筆者は従前領域情報の処理に関心があり、その処理手順に対して一案を有していた。これを基に専用処理システムを構想したものが文献 [1], [2] であった。その後この原理を用いた応用システムとしてグラフィックデ

ィスプレイ装置を製作し、前記専用処理システムが少くともこの応用分野では有効である事が確認出来た。この領域情報処理システムは、これまでのところは宇宙・特殊環境工学の立場で研究されて来たものではなく、又その基本思想自体に多分に未整理の部分を含んでいるが、この分野に一つの素材を提供し得るのではないかと考え、今回発表する次第である。

## 2. 領域情報

本文中に「領域」或いは「領域情報」と言う用語が多用されているが、ここに筆者なりの定義を記しておく。まず説明を簡単にするため対象とする空間を2次元に限定しておく。或る拡がりを持つ空間  $S(x, y)$  の中で、或る共通属性を有し、他と区別して認識することが可能な部分を「領域」と称しておく。つまり空間  $S$  の一点  $x, y$  はその空間の見方によって種々の属性(物理的特性, 社会的状態……等)を持っているが、その何れかに着目して切り出した部分空間である。

ここで夫々の領域について領域規定関数或いは領域規定情報  $r(x, y)$  を定義しておく、 $r$  は  $S$  に含まれ、

$r(x, y) = 1$  (算術式の場合) 又は = 「真」 (論理式の場合) である部分をその領域の内側と定め、

$r(x, y) = 0$  又は = 「偽」 の場合は外側とする。

ここで領域  $R$  に関する領域情報  $I_R$  は

$$I_R(r_R; a_{1R}, a_{2R}, \dots; C_{1R}, C_{2R}, \dots) \quad (1)$$

のように表わす。ここに  $r_R$  は前述の領域規定情報、 $a_{iR}$  は領域中に分布して定義された属性項目であり、 $r_R(x, y) = 1$  であるような  $(x, y)$  の組に対して  $a_{iR}(x, y)$  の値が定義される性質のものである。 $C_{iR}$  はこの領域全体を集約して定められる属性値である。簡単な例を挙げてみよう。空間に厚みの不均一な板を置いたとする。今この板の全体を一つの領域として識別するとする。その実体から情報化のプロセスを今は問わないとすればその  $r_R$  は既に与えられていることになる。 $a_{iR}$  としては板の各点の厚み分布、或いは表面の硬さ等が考えられ、これらを集約した属性  $C_{iR}$  としては面積、体積、重心、慣性モーメント、表面平均硬度、平均厚さ等がある。又社会的統計量の様なものが対象であれば、例えば  $r_R$  として「東京都」と言った既知の領域を指定すれば、 $a_{iR}$  としては人口分布、単位面積当り年間降雨量があり、 $C_{iR}$  としては総人口、年間総雨量などがあり得る。もしこの二例を汎用電子計算機で処理する場合を想定すると、夫々性質の異なる情報要素は別個の過程を経て最終的にはメモリ内の二進コード化された電子的情報としてまとめられ、処理に臨むことであろう。このように領域情報は最終的な処理の部分で、そこでの処理手段に則した実体を持たねばならない。しかし領域情報はその情報量が多いことが一般的な特徴であろう。したがって出来るだけ原情報のままで置くことに努め、処理の近傍でこれを変換することが出来れば実用上の価値が大であろう。

次に領域規定情報  $r_R(x, y)$  についてつけ加えれば、この値が1から0等に変化(反転)するところがこの領域の境界であり、この意味から  $r(x, y)$  は境界情報とみることも出来

るのである。概念的にとらえればこのように  $r(x, y)$  は領域拡がり情報、領域境界情報のどちらとしてもよいのであるが、後述の処理の段階ではその取り扱いに大きな差があるため、領域を規定する情報がその境界に関する情報によるとき、これを  $B_R$  と書くことがある。実例を挙げれば、境界上に適当間隔でとられた点の座標値データ群がこれであり、型板のテレビカメラによる直接走査などがこれまで述べて来た領域規定情報による定義例である。これらを処理システムの中に一時貯える場合、一般の  $r_R$  データの場合はこれをビットパターンにしてメモリに記憶し、 $B_R$  の場合は前述の如く座標値データの組として記憶することになろう。ディジタル処理を行うための離散化に伴う誤差の問題とその情報量との兼ね合いを考えれば、量子化の単位に比して大きい領域ではあるが、その形状が簡単なものには  $B_R$  の形態が、逆に小さいが複雑な形状のものには  $r_R$  の方が良いと推測される。領域情報の取り扱いに柔軟性を持たせるには、この両種の情報を目的に応じて自由に使い分ける様考慮しておくべきであろう。

本論文で想定する処理システムはディジタル回路技術をその基礎とするものであるので、領域情報は終局的には離散値データの集合で対象の原領域を近似することになる。この離散化に伴う精度等の問題は、本論文の範囲外であるが、後の応用化の段階における各種の検討はすべて現行システムとの比較が前提となっており、個別の検討はその都度為されていると言えよう。

### 3. 領域情報処理

前述のごとき領域情報の定義に基いて、本文における領域情報処理を以下の様に表現したい。

$$I_{RX} = P\{s\} \cdot [I_{R1}, I_{R2}, \dots, I_{Ri}, \dots, I_{Rn}] \quad (2)$$

ここで  $I_{RX}$  及び  $I_{Ri}$  は夫々(1)のごとく定義されるが、実際の領域情報は、領域規定情報  $r_{Ri}$  (又は  $B_{Ri}$ )、分布属性情報  $a_{Ri}$  及び集約属性情報  $C_{Ri}$  のすべてを含むとは限らず少くともこのうちの一つから構成されているとして取り扱う。  $P$  は処理全体を、又  $s$  はその操作内容を表わし、これを換言すれば一つ又は複数の領域情報を入力して新たな領域情報を生成する操作の集まりである。結果或いは出力としての  $I_{RX}$  については、その目的により  $r_{RX}$  のみが必要な場合もあるであろうし、又或る条件によって規定される  $r_{RX}$  内で或る定められた手順により属性情報を求める場合もあろう。一つの特例として、

$$I_{RX}(r_{RX}; a_{RX}; C_{RX}) = P \left\{ \begin{array}{l} r_{RX} = r_R \\ a_{RX} = \rho a_R \\ C_{RX} = g \iint a_{RX} \end{array} \right\} \cdot [I_R(r_R; a_R)] \quad (3)$$

の如き場合がある。これは領域規定情報が両者に共通であり、つまり入力領域情報の未知属性を求める処理例である。先の型板の例をとり挙げれば、既知情報  $a_R$  として板厚を考え、 $a_{RX}$  は分布質量、 $C_{RX}$  は重量である。

しかし一般に求める領域は入力領域とは別のものであり、その領域規定情報は、入力領域

情報の要素と与えられた定数とにより定義される性格のものである。例えば領域  $R1, R2, R3$  の共通部分で且つ  $R4$  に含まれない部分を新領域  $RX$  とする場合は

$$RX \leftarrow R1 \wedge R2 \wedge R3 \wedge \overline{R4} \quad (4)$$

と書くことにする。これは新領域を規定する条件が

$$r_{RX}(x, y) = r_{R1}(x, y) \wedge r_{R2}(x, y) \wedge r_{R3}(x, y) \wedge \overline{r_{R4}(x, y)} \quad (5)$$

である事を表わしたものである。

別の領域規定の方法として、或る特定の属性値を持つ部分を切り出すことも考えられる。 $a_R$  を入力情報に含まれた一つの分布属性情報として

$$RX \leftarrow R(a_R = k) \quad (6)$$

とした如き例である。これは

$$r_{RX} = \{a_R(x, y) = k\} \quad (7)$$

であり、つまり  $a_R(x, y) = k$  が真であるとき、 $r_{RX}$  も真となりその領域を規定する。更に一般的な操作はこの両者を複合したものであり、下記はその一例である。

$$RX \leftarrow R1 \wedge R2 \wedge R3 \vee R4 \wedge R5 \wedge R\{(a_{1R} = k_1) \wedge (a_{2R} = k_2)\} \quad (8)$$

これは新領域の規定に属性情報が用いられる場合を示したものであるが、逆に新領域の属性情報に領域の規定情報が用いられることがあってもよい。即ち

$$a_{RX} = a_1(x, y) \times r_1(x, y) + a_2(x, y) \times r_2(x, y) \quad (9)$$

の如き例である。(3)式の{ }内は(2)式の  $s$  の内容であるが、ここで

$$\left\{ \begin{array}{l} r_{RX}(x, y) = r_{R1}(x, y) \wedge r_{R2}(x, y) \wedge \dots\dots\dots \\ a_{RX}(x, y) = a_{R1}(x, y) + a_{R2}(x, y) + \dots\dots\dots \\ c_{RX} = K \left[ \iint_{RX} \{a_{R1}(x, y) + k a_{R2}(x, y) + \dots\dots\dots\} dx dy \right] \end{array} \right\} \quad (10)$$

のように異なる領域の構成情報間での演算が処理の一般形態であろう。上例は各領域情報が同一の座標系で定義されている特殊な例とみることも出来よう。つまり一般に原情報を処理システムの入口でみた場合、これらがすべて同一の座標系上で定義されているとは限らないからである。しかし領域情報の処理が複数領域相互の関係をもとに一つの領域情報を創り出すことにあるので、これら領域の幾何学的関係は、処理前定義、処理結果による事後判定等の別はあるにせよ、とにかく処理時には一義的に定まっている筈である。そこで本文では入力情報の事前処理段階に必要な座標変換処理が含まれている事を前提として常に(10)式の如き表現を用いることとする。(10)式を処理手順と言った観点からみると、まず  $r_{RX}$  の演算を行い、これが真のとき、つまり求める領域の内部であるとき、以下の演算を行うこととなる。この演算を或る定められた空間の全域にわたって実行する訳であるが、その空間

は処理対象領域とでも称すべきものであろうか。しかしここでは専用処理システムとしてディジタル回路を基盤としたものを念頭に置いている。したがって対象の二次元領域は或る小領域に分割され、領域情報は離散系データの集まりとして取り扱われる。演算は対象とする全体空間に含まれるすべての小領域のこれら離散データの夫々について、或る定められた順序のもとに所定の処理を施すことであるから、今後この全体の空間を走査領域と呼ぶことにする。

以上の考えをまとめると領域情報の処理は次の過程より成る。

- (1) 原情報より入力領域情報を生成。  
(座標変換を含み、以後の走査が可能な形態に変換する。)
- (2) 走査空間、走査手順の決定。  
(量子化の程度等もこの段階で定まる。)
- (3) 走査座標の指示。
- (4) 上記座標における入力領域の規定情報、属性情報を出力する。この情報を以後局所領域情報と呼ぶことにする。又この処理段階を走査処理と称するが、各個別領域毎に独立に行う処理でもあり、単位処理と呼ぶのも一つの見方であろう。
- (5) 上記局所(入力)領域情報間に所定の演算を実行する。この過程を総合処理と称する。
- (6) 走査領域全域にわたるよう、(3)、(4)、(5)の操作を繰り返し、所要の結果(最終出力)を得る。この過程は呼称上は総合処理に含める。

以上を図にしたのが第1図である。走査方式の一つとして座標変換を走査座標側に施すことも考え得る訳であり、その場合は各入力領域情報に与えられる  $x_c$ 、 $y_c$  は互いに異なる値を持つものとなる。又走査に規則性がある場合は、 $x_c$ 、 $y_c$  をその都度与える必要は無く最初に一括してパラメータを与え、以後は処理の同期を保つための情報のみを与えることでもよい。従ってこれらを一括して表現する場合は走査情報という言葉を用いる。出力としての領域規定情報  $r_{RX}$ 、分布属性情報  $a_{iRX}$  については第1図中の局所(出力)領域情報を蓄積すればよいが、集約属性情報  $C_{iRX}$  については逐次的に或いは一括して走査領域全体における出力を対象にして処理を実行する。集約属性処理の簡単な例を挙げてみる。

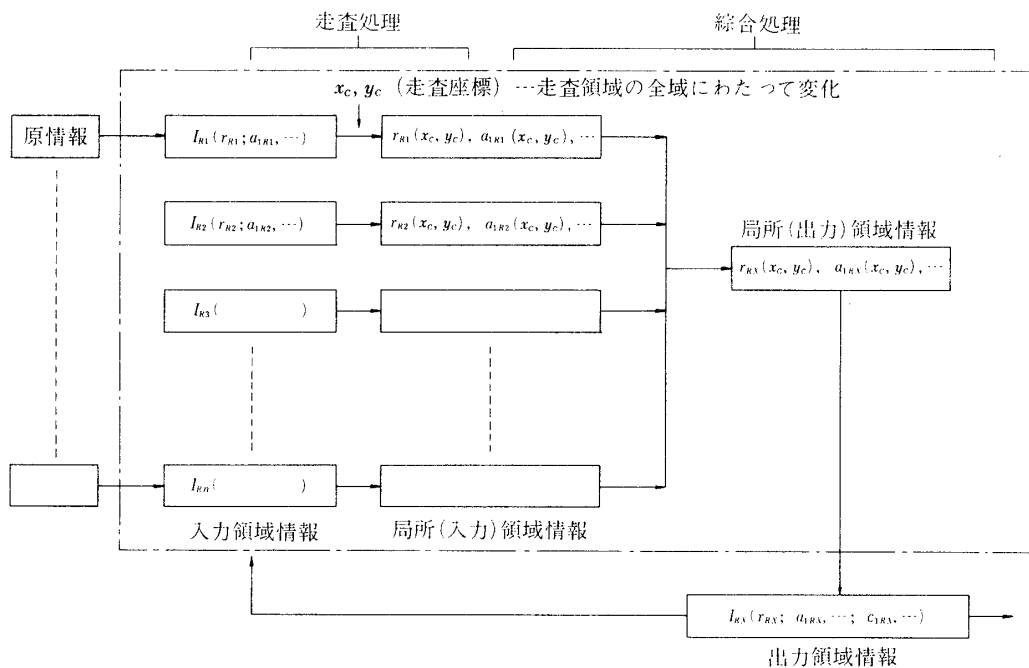
- (1) 「求める条件の領域が存在するか否か？」のみが問題のとき。先づ  $C_{RX}$  を「存在せず」の状態にして走査を開始、 $r_{RX}(x_c, y_c)$  が真のときこの  $C_{RX}$  を「存在する」にする。
- (2) 面積を求める。 $C_{RX}$  を面積とし、初期値を0として走査を開始、 $r_{RX}(x_c, y_c)$  が真のときこの  $C_{RX}$  へ定数を加算してゆく。最終的にこの  $C_{RX}$  の示す値が求める領域の走査領域内に含まれる部分の面積に相当する。

なおここで得られた出力領域情報を次の入力領域情報とすることも可能であり、これにより繰り返し処理が可能となって来る。

#### 4. 専用処理システム

##### 4.1 装置構成の基本方針

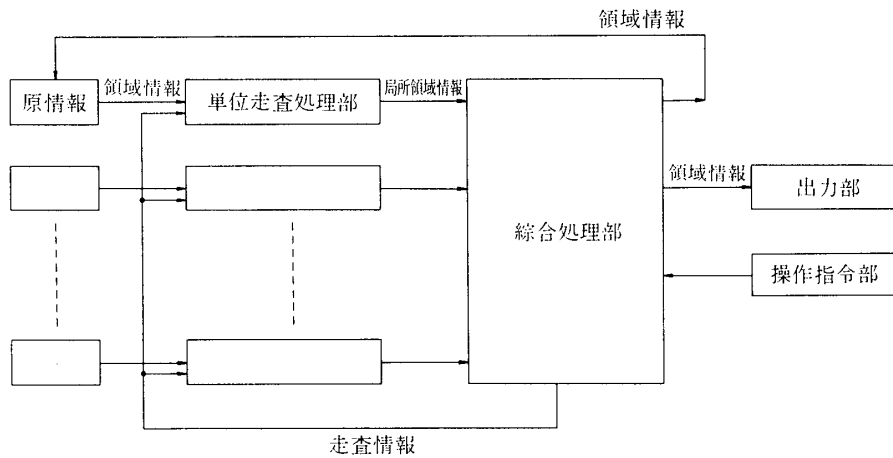
先に領域情報の処理を第1図の如く整理した。これを現在の汎用電子計算機で実行するこ



第1図 領域情報処理の流れ

とは勿論可能である。原情報を夫々その性格に適した入力系を通じてシステム内に取得し、メモリ内に格納する。これを逐次取り出して所定の処理を実施するのであるが、第1図の方式は元来電子計算機の得意とする四則演算で処理出来る部分をわざわざ多数点データの相互演算に置き換えた様なものであるから、現在、既に計算機で実行されている種類の処理においては不利な点が目立つことであろう。やはり領域情報処理の問題をこの様に整理した以上、先ずこの方式に適した専用システムを考え、従来の計算機システムにとって不得手であった問題に対する特性を吟味し、その上で従来の計算機で実施されて来た問題についての比較に進もうとの基本方針をたてたのであった。

第1図に明らかな如く、単位毎の走査処理つまり単位走査処理は各入力領域情報毎に独立して実行出来るのであり、共通の走査座標に応じて局所領域情報を出力するという条件を満たしていれば良い。従って、この問題を装置上のインターフェースと言う観点からみれば、統一した制御下にある走査関係の信号を受けとり、同一仕様の局所領域情報を出力するように構成された単位走査処理部を必要数並置すればよいことになる。総合処理はこれらの局所領域情報を受けて最終結果に至る処理を行う部分であり、システムとしては、これを一つの機能単位として扱うことが出来よう。これを総合処理部と称することにする。これは更に装置利用者からの指令を受ける部分であり、又走査情報を送り出す部分でもある。更に一部の原情報はこの総合処理部を経て設定されるであろう。この機能は出力情報を再度入力情報として利用するためにも必要であろう。以上の概念を示したのが第2図である。図中単位走査処理部は取り扱うべき入力領域情報の組の数に見合うだけ設けられるが、処理速度は一般に、



第2図 領域情報処理システム

その数とは無関係である。これをハードウェアの観点から見るとインターフェースとしての条件が少ないので可成り自由な構成が期待出来る。もしすべての単位走査処理部に共通の走査様式が適用出来るのであれば、原情報として、実景、空間内の電波反射体、水中の音波反射体、型板等をメモリ内の情報と同列に扱うことが可能となる。つまりこれらをテレビカメラ、レーダ、ソナー、機械的プローブ又はテレビカメラ等で走査し、同形の局所領域情報を得、これを総合して処理することが出来る事となる。原情報から局所領域情報を得る過程が短い事、つまり途中に介入する段階が少なくて済むと言う事はシステムにとって有利な材料の一つである。これらの試作意欲をそそるテーマもあったのではあるが、システムの具体化の第一段階としては、あえて電子論理情報型の原情報を主たる入力とするものに絞る事とした。これは後述のディスプレイ装置への応用が第一目標であったからに他ならない。このように限定した場合、第2図の原情報の媒体はメモリでありその内容は、ビットパターンとして表わされた領域規定情報 $r_R$ 又は座標値データとしての境界情報 $B_R$ 、それに分布属性情報等である。

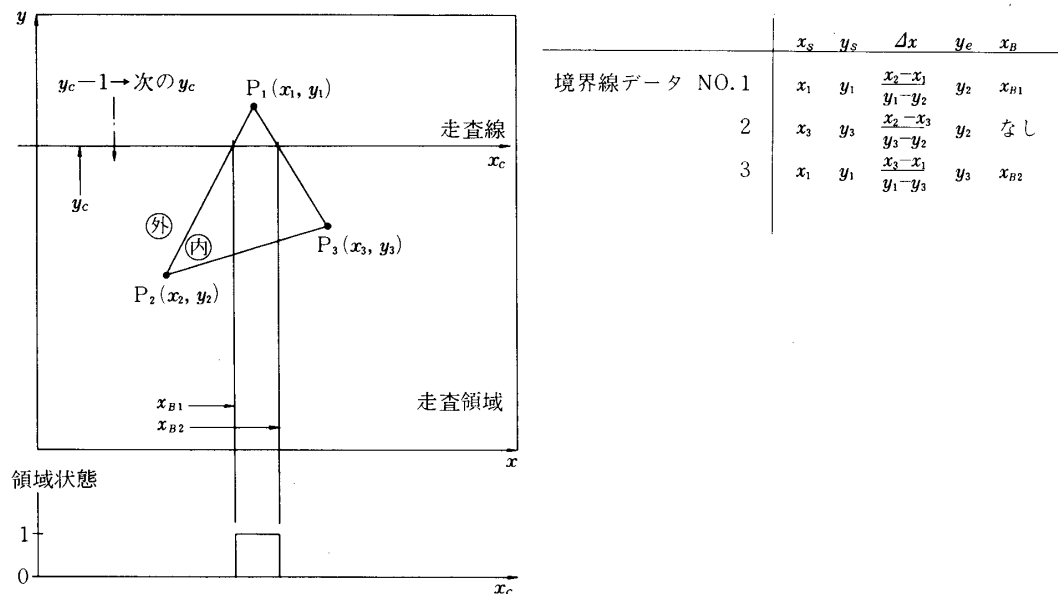
#### 4.2 単位走査処理部

この単位走査処理部の構想を具体化してゆく場合の最大の問題は境界情報 $B_R$ の取り扱いであった。境界点の座標データから一旦領域規定情報としてのビットパターンをメモリ内に作り出し、これを走査順に出力してゆくと言うのが従来の手法であろうが、この形の原情報の量が多い場合を想定すると、ここが高速処理に対する一つの隘路となる事が予想されたのであった。この問題は例えばグラフィックディスプレイ装置で多くのベタ塗り図形から成る画面を1フレーム毎に書き替える時などに発生する。以上の事情があり、境界情報 $B_R$ そのものが領域規定情報 $r_R(x, y)$ と全く同等に扱える装置を求めてみる事となったのである。

先ず走査様式は、線に沿って行われる線状走査の、線位置を順次移動させて行う面走査を考える。これまで走査の順序又は並行処理か逐次処理かの別は問題にしていなかったが、ここでは一応前記の如き限定が行われた訳である。

第3図は、 $P_1 P_2 P_3$ の三角形領域の例である。今走査はテレビ方式とし、 $x$ 方向は右向き走査を行ない、 $y$ 座標は大きい方より小さい方へ、つまり下向きに移動させる。3点の座標データを一旦図中に示した形の境界線データに変換する。これは上端点の $x$ 、 $y$ 座標 $x_s, y_s$ 、一回の走査線の移動量に対応する $x$ の変化分 $\Delta x$ 、下端点の $y$ 座標 $y_e$ 及び走査線との交点座標 $x_B$ を一組とするデータである。或る水平走査が行われる直前にその走査線の $y$ 座標 $y_c$ は与えられている。ここで或る一つの境界線に着目する。面全体の走査を開始する直前の $y_c$ は、すべての境界線の上端点より上側にあるか一致していると仮定すれば、最初は $x_B$ は「なし」の状態、次に $y_c = y_s$ となった時点で $x_s \rightarrow x_B$ 、以後 $y_c = y_e$ まで線走査が一回終了する毎に、 $x_B + \Delta x \rightarrow x_B$ とし、 $y_c = y_e$ のとき $x_B$ を「なし」にする。次に領域状態値と言うものを考える。これは領域の内側で1、外側で0の値をとるものとし、 $1 \rightarrow 0$ 、 $0 \rightarrow 1$ の変化を領域状態の反転と呼ぶ。そこで走査線左端での値を領域状態初期値と称しておく。ここで走査面全体を考えるとその走査開始点、つまり走査領域の上左端の状態は予め与えられている必要がある。言わば領域状態初期値の初期値である。走査線の下降に応じて走査領域の左縁を越えるデータがあったとき、この領域状態初期値を反転させる。

さてここで一つの線状（水平）走査に着目する。先ずこの時点での各境界線についてそれらの交点座標 $x_B$ をその大きさの順に取り出せる様な状態にしておく。走査開始点で領域状態値には前回の走査の後に確定している初期値が入れられる。以後 $x_B$ を値の小さい方から



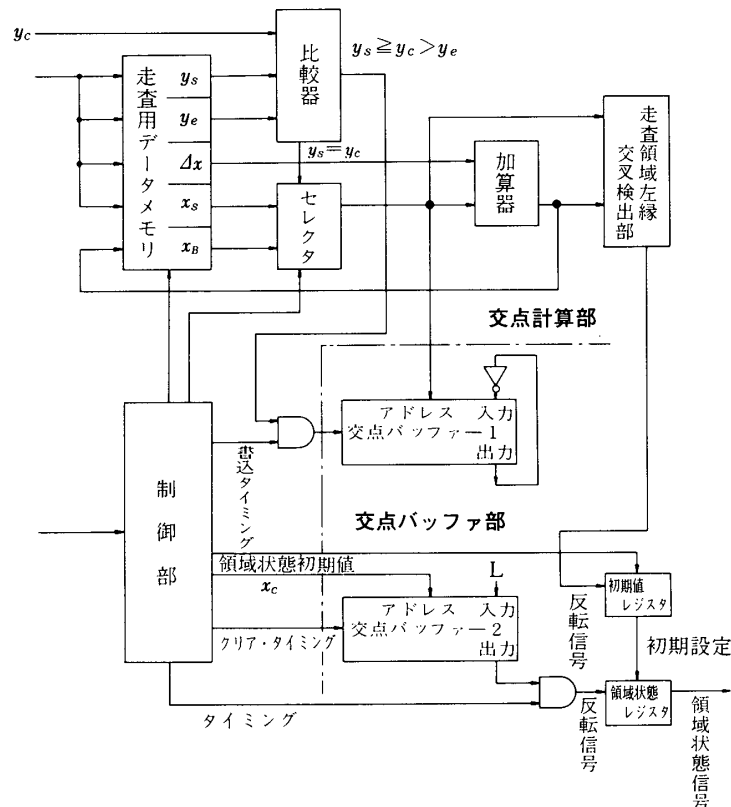
第3図 境界線データの走査



順次取り出し  $x_c$  (走査  $x$  座標) =  $x_B$  の点で領域状態を反転させる.  $x_B$  「なし」のデータ及び走査領域の右側にはみ出た分は影響がない. 同様に下側にはみ出る分も考慮する必要がない. 境界線が水平の場合も  $x_B$  は「なし」としておけばよい. その左右端を端点とする他の二本の境界線データが領域状態値を正常に制御してくれるからである.  $P_1, P_2$  点では反転を同時に 2 回生じさせるのが最も簡単な走査方法であり結果的に状態値は変化しない. その他実際問題としては種々特別の場合の対応を考慮する必要があるが, そのほとんどは境界線データ作成の過程で処置出来ると判断した. ここで得られた領域状態値を定められた規則に従って出力したものが局所領域情報としての領域規定情報  $r(x_c, y_c)$  である. 以後装置の説明としてこれを領域状態信号と呼ぶ場合もある.

以上述べた如き機能を実現する装置が作られれば, それが境界情報  $B_R$  から直接局所領域規定情報を出力する単位走査処理部となるのである. これに対する具体案にはソフトウェア色の強いものから純ハードウェア的なものまで種々の形態のものがあり得るが, 現時点で最も実用性が高いと考えているものを以下に紹介する.

第 4 図はその概略を示したものである. 処理は二期に大別される. 第一期での処理は, 予



第 4 図 境界線データ走査回路

め走査用データメモリ中に格納されていた  $n$  ケの境界線データを読み出し、もし走査線との交点  $x_B$  が存在するのであれば、その  $x_B$  をアドレス値として交点バッファに交点の存在位置を記録することである。第二期ではこの交点バッファに走査仕様に従って順次変化するアドレスを与え、交点存在の記録が読み出された時点で領域状態レジスタの出力を反転させる処理を行う。第二期の長さは一水平走査周期に等しく、第一期も同様にとるとすれば結局一水平走査分の処理を二週期かけて行うことになっている。

第一期において、走査用データメモリ中の  $y_s$  等  $n$  組の境界線データのうちの一组が読み出されると、そのうちの  $y_s, y_e$  が比較器で  $y_c$  と比較され、 $y_s \geq y_c > y_e$  のとき、つまり境界線と走査線が交叉するとき、そのときの  $x_B$  ( $y_s = y_c$  のときは  $x_s$ ) をアドレスとして交点バッファに書込みが行われるが、それ以外では書込みは行われず、これが  $x_B$  「なし」に相当する。図中セレクトは  $x_s$  と  $x_B$  の選択読出機能を代表させたものであり、 $y_s = y_c$  となった時のみ  $x_s$  側を読み出し、それ以後は  $x_B$  側が読み出される。一方加算器により  $x_B$  (又は  $x_s$ ) と  $\Delta x$  とが加算され、その出力が  $x_B$  メモリ部の入力となっている。従って  $y_s \geq y_c > y_e$  の間  $x_B$  の値は常に更新され、その時の走査線との交点座標を示すことになる。

第一期の終了時点において  $y_c$  の値が更新され、交点バッファ - 1, - 2 が交換される。次の第一期操作はこの交換された交点バッファを対象として、更新された  $y_c$  データに基いて実行されるが、これと並行して先に書き込まれたバッファを対象として第二期の操作が行われる。

第二期では、制御部から走査座標  $x_c$  に相当するアドレスを順次与え、その出力により領域状態レジスタを制御する。これにより交点位置に至る度に領域状態信号が反転する。

なお、書込側の交点バッファでは、入力信号として出力信号を反転させたものが与えられている。これは書込直前の内容を反転させる事を示したものであり、こうすることにより、同一点に偶数箇の交点が重なったときは、交点が存在しなかったことに等しい効果を得ている。尚このため第一期の開始前に交点バッファの全内容が 0 となっている必要があるが、図中交点バッファ - 2 の L 入力とクリア・タイミング信号はこのためである。ただし同一期に読み出しとクリアの両方が実行出来ないときはクリア期を別箇に設ける。したがってバッファは三面切換制御となる。

領域状態初期値の反転条件の検出法は走査用データの作り方と密接な関係がある。図の場合は走査用データのフィールドが走査領域左はみ出し部の負の座標値も表現し得る場合の例である。

この第 4 図の構成によるときは境界線は直線に限られるが、 $\Delta x$  を多段にすることにより曲線を扱うことも可能である。

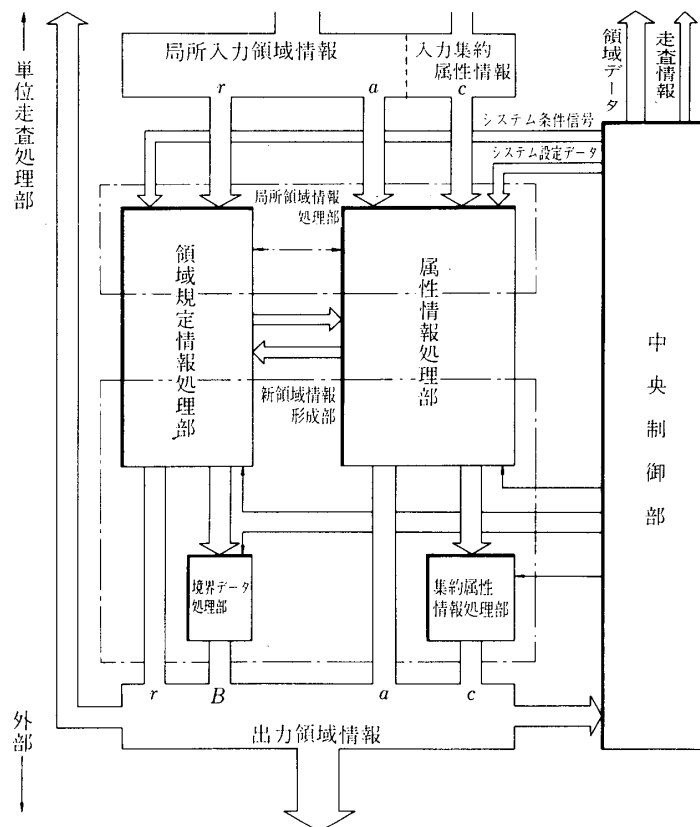
一方の分布属性情報の取り扱いに関しては、原データの座標系の変化に迅速に対応出来る柔軟なアクセス方式が必要であるが、一応現用技術で対処出来る部分が多いので今回は割愛する。

#### 4.3 総合処理部

この総合処理部の機能は、一つ又は複数の領域情報を走査して得られた局所領域情報を入

力として、これを処理して所要の新局所領域情報を生成すると共に、必要に応じて全走査領域に亘っての集約的処理を行い、全体として少くとも一つの新しい領域情報を出力することである。想定するシステムの目的に応じて単能的なものから大規模汎用システム用の汎用総合処理部まで多様な形態が考えられるが、これを一般化して表わしたのが第5図である。

入力はず局所領域情報処理部において主として同種データ相互間の演算処理が行われ、次段の新領域情報形成部で、出力領域情報に関する処理が行われる。一方データの種類のに応じて、つまりはハードウェア構成上の別から区分すれば、図の如く論理演算を主体とする領域規定情報処理の部分と算術演算を主とする属性情報処理部とに分けることが出来る。領域規定情報処理部は(8)式の如き論理演算を行うので、そのハードウェアとしてはPLA素子の如きAND-ORマトリクス回路で且つその交点の接続状態が随時変更出来る型式のものが適している。(8)式の中には属性値による項があるが、これに相当する信号は属性情報処理部の局所領域情報処理部で生成される。一方属性情報処理部は(9)式の如き属性値に関する演算を実行するが、これに必要なデータの一部は領域規定情報処理部の局所領域情報処理部より供給される。又集約属性処理部はこの属性情報処理部の出力を処理する部分として位置



第5図 総合処理部

づけてよいであろう。尚(8)式の  $k_1$ ,  $k_2$  等入力領域情報に含まれないデータは図の如くシステム設定データとして別箇に与えられる。同様に領域規定情報(領域状態信号)と同等の機能を持つシステム条件信号も有用であろう。中央制御部は各処理部を制御するが、汎用型処理部では内部処理手順も制御対象に含まれるであろう。これには、例えば領域規定情報処理部の AND-OR マトリクス回路の交点接続状態の制御なども含めて考えている。

出力領域情報は、直接外部へ出力されるものの他に、単位走査処理部へ送られるものもあってよい。又中央制御部はこの出力領域情報を帰還情報として参照して、システムに所要の変更を加えて次の処理を行う等の自動繰り返し処理も考慮すべきであろう。

第5図は3.で述べた領域情報処理のすべてが行える様画かれているが、もし処理の目的が限定されたものであれば、当然第5図の構成要素の一部を省くことがあってもよい。例えばテレビカメラで図形を撮し、その面積を求める最も単純なシステムでは、集約属性処理部を単なるカウンタとし、テレビカメラと結ばれた単位走査処理部からの唯一つの領域状態信号をこれに直接接続すればよい。他の処理部は不要である。

夫々の処理部及び中央制御部の構成を如何にすべきかは、そのシステムの目的に応じて検討すべき問題である。例えばこれらのすべてを現在の電子計算機技術を中心とするものでまとめることも可能である。特に属性情報処理部と集約属性処理部或いは中央制御部は、いわゆる「計算」処理を行う割合が多く、計算機技術との関連が深いであろう。これとは逆に領域規定情報処理部は多数の1ビットデータ間の論理演算を行うところであり、専用ハードウェア化の利点の多い部分である。この部分を主要な構成要素とするようなものに例えば多数の動く目標を監視し、或る特定領域内にあるか否かのリアルタイム検出システムの様なものが考えられる。

中央制御部は、走査全体の制御も行う。このためこれに関する信号を単位走査処理部に送り出す。この他領域情報の管理、領域情報の前処理(座標変換等)を受け持つ場合も考えられる。ただしこの規模が大きい場合は、別箇の独立処理部として扱うべきであろう。

以上を総括すると、ここに想定したシステムとは、少なくとも1個の単位走査処理部に対して設定した領域情報と、総合処理部に対して指定した処理方法とによって新たに所要の領域情報を生成するためのものである。

## 5. 応用システムとしてのグラフィックディスプレイ装置

ラスタスキャン型のグラフィックディスプレイ装置は管面上の表示範囲を走査領域として、そこに或る表現意図のもとに制御された属性情報を持つ一つの領域情報を出力させる装置と考えることが出来、一つの領域情報処理システムである。

筆者は昭和51年頃よりグラフィックディスプレイ装置を中心とするロケットの管制システム開発に関与するようになったが、汎用のディスプレイ装置ではなかなか思う様な表現が出来ず、そこで領域情報処理システムの観点からこの種の装置を吟味し、より目的に合う装置を探り出そうと考え、検討を試みたのであった。この時必要とされていたのは、コンピュータグラフィックスの部類に属する表示を、より細やかな表現で、より高速に行うシステムであり、これはアニメーション画のリアルタイム生成システムを求める事と同質と考えてよ

いであろう。この場合、画面の設計者は表現したい事項を適宜表示図形の形、大きさ、位置、向き、色、輝度等に託し、これを幾つか組合わせて全体を構成してゆく。この最小の要素を表示単位と呼ぶ事にする。例を先のロケット管制システムにおける一つの監視用画面にとれば、ここでは、軌道の左右ヅレを横軸に、姿勢の横方向への傾き誤差を縦軸にとって画面が構成されている。ここに表示される誤差の許容範囲を示す枠は一つの表示単位であり、ロケットの現在状態を示すマークも表示単位の一つであろう。枠の表示単位はロケット発射後の経過時間に応じて予め定められた形を表わす必要があり、後者はレーダ、ジャイロからの観測値に応じて表示位置を動かさなければならない。これと共にその観測系側に異常が見出された時には、形又は色を変える等の処置が要望される。このように各表示単位は個別に制御される必要がある。

以上の例を、前述の領域情報処理の一般論にあてはめてみると、各表示単位は入力領域情報に相当し、その表示単位が固定した表現（色等）を持つ時は、それをその領域情報の属性情報と考える。又表示単位の輪郭は境界点データによって与えられる事が多いであろう。表示画面は複数表示単位を複合して得られるが、何も表示しない部分も黒色背景と言う表示単位であると考えれば、表示出力としての領域情報では、表示範囲そのものが領域範囲であると考え常に  $r(x, y) = 1$  として取り扱う事が出来よう。

ここで、ディスプレイ装置としての性格を特徴づけるものは、総合処理の中の分布属性情報を形成する処理部分であろう。ここに意図しているアニメーション的画面は、各種の形と色の型紙を重ねてそれを上から眺めることに例えられるが、この場合表示の必要度の高いものが上側へ来る様に、その上下順位を定める事であろう。又その中には透過性を有するものと、不透過性のものとが混在するのが一般的な姿であろう。この時表示画面中の一点に着目すれば、そこに同時に存在する表示単位のうちで不透過性のものについては一番上側のものが表示され、透過性のものについては、この最上位不透過性表示単位より更に上側に定義されているものが表示される。この様に一つの不透過性表示単位に対応する属性値出力と、これより上側に定義されている透過性表示単位に対応する属性値出力とを合成したものがその点の表示用出力となる。(9)式の形でこれを表わせれば以下の如くである。

$$a_{RX} = a_{ei} \times [a_{zi} = N_z(1, N_q)] + \sum_j a_{ej} \times [a_{zj} > N_z(1, N_q)] \times r_j \quad (11)$$

$$N_z(1, N_q) = M_{ax} \{ a_{zi} \times r_i \} \quad (12)$$

$a_{ei}$  : 不透過性表示単位に対応する表現値 (RGB 出力等)。

$a_{ej}$  : 同じく透過性単位に対応するもの。

$a_z$  : 表示の上下順位を定めるための属性データ

$N_q$  : 不透過性表示単位の数

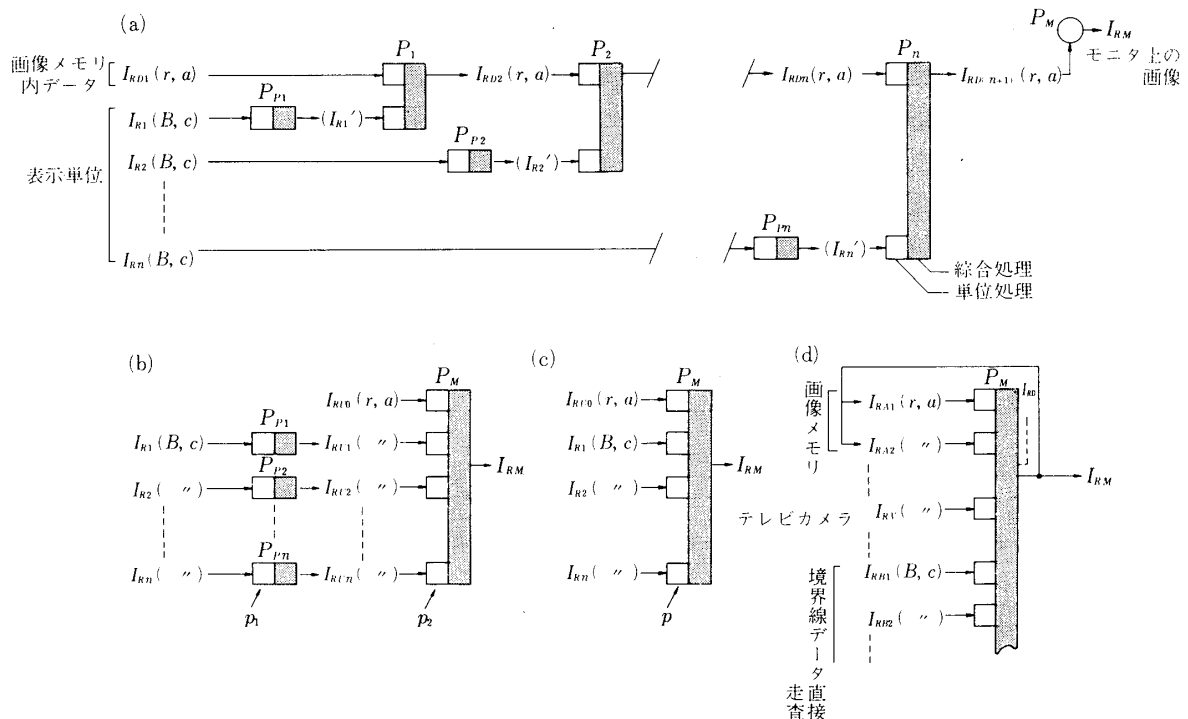
$r_i, r_j$  : 領域規定情報

[ ] : 内部の論理式の値が真の時 1, 偽の時 0 の値をとる。

以上により  $N_z$  は  $r_i = 1$  である表示単位の  $a_{zi}$  の最大値であり, (11) 式の右辺第一項は最上位不透過性表示単位の出力, 第二項は更にこれより上側にある透過性表示単位の出力である. ただしこの (11) 式では各出力を単に加算しているので実際の光学的透過現象を再現している訳ではない.

次に, 現状の一般的なディスプレイ装置で実行されている処理を 3. で述べた領域情報処理の観点から整理し, その上で改善策を見出してゆく事とする. 第 6 図(a)は画像メモリを 1 面のみ備えた通常型ディスプレイを想定したもので, 先ず背景に相当する  $I_{RD1}$  が画像メモリ内に作られているとする. (黒地であれば全メモリをクリア) これは領域規定情報  $r$  と分布属性情報  $a$  とより成る領域情報の例である. 但し前にも述べたように  $r$  については常に 1 であるとしてよいであろう. 各表示単位についてのデータ  $I_{Ri}$  は, まず境界点データ  $B$  と内部の色  $c$  との組み合わせで与えられている. これを  $I_{R1}$  から順番に画像メモリに書き込んでゆくが, その前に処理  $P_{P1}$  によって  $(B, c)$  の形で与えられているデータを  $(r, a)$  の形に変換する. この結果  $a_{R1} = r_{R1} \times c_{R1}$  となり, これは画像メモリに書込可能な形態である. 処理  $P_1$  は  $r_{R1} = 1$  であった箇所のデータを  $a_{R1}$  に置き替える. これを (3) 式にならって式で表現すれば

$$I'_{R1}(r_{R1}; a_{R1}) = P_{P1} \left\{ \begin{array}{l} r_{R1} \leftarrow B_{R1} \\ a_{R1} = c_{R1} \end{array} \right\} \cdot [I_{R1}(B_{R1}; c_{R1})] \quad (13)$$



第 6 図 グラフィックディスプレイ装置データ処理の流れ

$$I_{RD2}(r_{RD2}; a_{RD2}) = P_1 \left\{ \begin{array}{l} r_{RD2} = 1 \\ a_{RD2} = a_{RD1} \times \overline{r_{R1}} + a_{R1} \times r_{R1} \end{array} \right\}.$$

$$[I_{RD1}(r_{RD1}; a_{RD1}), I'_{R1}(r_{R1}; a_{R1})] \quad (14)$$

となるが、 $\overline{\phantom{x}}$  は 0, 1 の 2 値間での反転を意味している. これを  $n$  回の表示単位について逐次的に実行し, 最終的に画像メモリ内に  $I_{RD(n+1)}$  を得る. これをモニタ上の視覚情報とするために,  $P_M$  と言う映像化媒体変更処理がある.  $P_{Pi}$ ,  $P_i$  処理は (ホスト) 計算機又はディスプレイ制御装置内の操作である. 表示単位相互の上下関係は, その処理順序によって定まるが, 透過性を実現するためには何等かの機能部追加が必要であろう.  $P_{Pi}$ ,  $P_i$ ,  $P_M$  の処理時間を夫々  $T_{PP}$ ,  $T_P$ ,  $T_{PM}$  とすれば, 全体の処理時間  $T$  は

$$T = n \times (T_{PP} + T_P) + T_{PM} \quad (15)$$

となる. 又この方式では画像メモリ中のデータから各表示単位を分離して識別する事は困難である. 従って如何なる小修正といえども全表示単位に影響を及ぼす可能性があり, もし全表示単位にわたる無駄な操作を避けようとするれば, 何等かの方法で変更の影響範囲を予測した上でその部分の処理を行わねばならない. これはソフトウェアの負担を大きくする.

この問題を解決するためには各表示単位に夫々画像メモリを持たせればよい事になる. 第 6 図(a)の場合には各  $P_{Pi}$  の出力  $I_{Ri}'$  に対して夫々メモリを与える事を意味するが, こうすることにより各画像メモリは独立に制御出来るので個別変更時の処理を簡単にするが, 変化する表示単位が多いときは, 投資が大きい割には効果の期待出来ない構成である. そこで各表示単位の画像データを画像メモリ中で複合するのではなく, 全部並行に出力させ外部で複合させる事を考えてみた. 第 6 図(b)がそれである. ここで  $I_{RU0} \sim I_{RU n}$  はすべて専用画像メモリ中の  $(r, a)$  型のデータである.  $I_{RU0}$  は先の  $I_{RD1}$  と同じく背景データである.  $(B, c)$  型, つまり境界点データで与えられた表示単位  $I_{Ri}$  は処理  $P_{Pi}$  によって  $(r, a)$  型のデータに変換される. その上で全画像メモリの内容が複合されモニタへ送られる. 出力合成部を全く新規に設計する必要はあろうが, 処理の高速化と部分変更の容易さとの両面にわたっての改善が期待出来る. 処理時間は,  $T_{PM}$  は不変として

$$T = n \times T_{PP} + T_{PM} \quad (16)$$

となる. 各  $P_{Pi}$  は並列に実行出来るのであるから, 処理装置をそのように構成する事により尚一層の高速化を企てる事が出来る. この場合

$$T = T_{PP} + T_{PM} \quad (17)$$

である. この図で明らかな如く各表示単位についての処理は最終的に出力が合成される直前まで全く独立に行われる. 第 6 図(b)の  $P_M$  処理の単位処理側の部分  $p_2$  は表示単位  $I_{Rn}$  のみに関する処理である. 従って, もし  $P_{Pn}$  の単位処理側  $p_1$  を  $p_2$  と同じ形にする事が出来れば, 画像メモリ ( $I_{RU n}$  部) を省き, その両側を直結する事が可能である. この場合を想定した

のが第6図(c)であり図中 $p$ が(b)の $p_1$ ,  $p_2$ に相当する。これは(b)と同じ処理特性を持ちながら画像メモリを背景分のみに局限した例であり、この背景にしても全面黒色等単純なものであれば、この分のメモリも不要である。処理時間についても

$$T \doteq T_{PM} \quad (18)$$

となる。この様なシステムが実現可能か否かの極め手になるのは境界点データから直接領域規定情報を出力し得るハードウェアが設計出来る事と、上下関係の制御が可能な出力合成ハードウェアが得られる事である。

現行システム第6図(a)と改良案(c)とを比較すると処理速度と動的表現の容易さとの点で(c)が明らかに有利である。しかし表示単位の数に見合うだけのハードウェア構成をとる必要があり、その意味から、複雑ではあるが準静的な画面に対しては(a)が有利である。

表示対象としているのが動画である事は先に述べたが、動画と言ってもその内容は非常に多様であり、一概にその性格を定める訳にもゆかないが、一般には少数の「複雑・準静的表示」とこれよりは多い若干数の「簡単・動的表示」の組合わせで大方の目的は達せられるのではないかと考える。この様な仮定のもとに第6図(c)に若干の修飾を加えたのが同図(d)である。 $I_{RA1}$ ,  $I_{RA2}$ , ... は画像メモリを媒体とする表示単位であり、場合によっては他の表示単位の内部の模様などにも用いられる。 $I_{RB1}$ , ... は境界点データとして設定する事の出来る表示単位である。この $I_{RA1}$ , ... に対しては出力 $I_{RM}$ 又は別箇の出力 $I_{RD}$ が還元される様になっている。(B, c)型の表示単位を組み合わせる背景画を作る様な場合この系統が利用出来る。又 $I_{RV}$ はその他の形態の走査処理部の例として挙げたものでテレビカメラである。これにより外景等を中央制御部の制御のもとに複合表示する事が出来る。

先に4.において処理システムについての一般論を述べたが、これからグラフィックディスプレイ装置の具体的な説明に入る前に若干これを補足したい。

先ず全体構成についてであるが、第6図における検討の結果最終的に得られた(d)の構成は、第2図の領域情報処理システムの一般形態に帰着した事になる。ディスプレイ装置としての特殊性は総合処理部内の属性出力データ形成部分のハードウェアと領域データの管理法にあるとみてよいであろう。

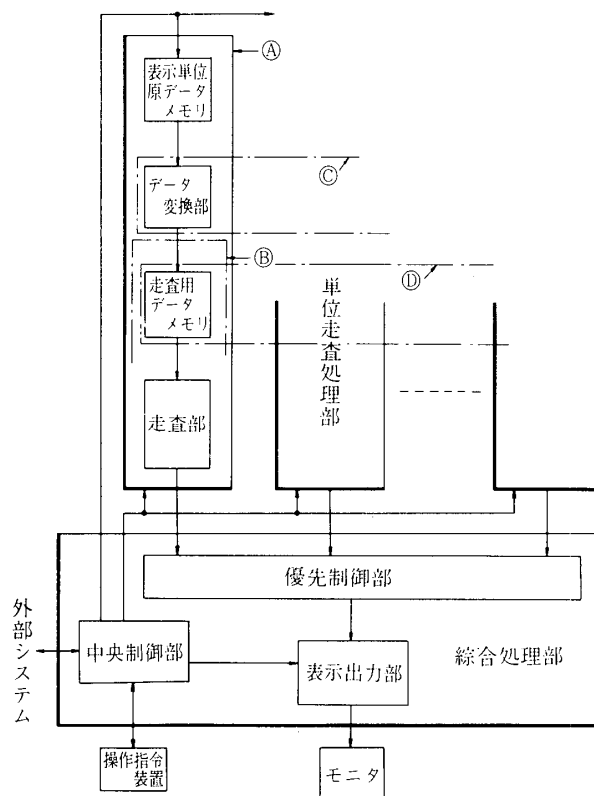
次に単位走査処理部についての補足事項を述べるが、ディスプレイへ応用する時の一つの鍵となる境界点データの直接走査法に関しては、第4図の如き方策があるので特に問題はないと判断された。むしろ従来型の画像メモリへのアクセス法に今後の努力が注がれると考えられる。その理由は、境界点型のデータについては、一旦その走査法が得られれば、表示単位についての座標変換を伴う動的変更操作は非常に容易であり、事前に要する処理の量が非常に少なく、このためこの形態の表示単位が高速に変化し得ることとなり、これが画像メモリ型の表示単位に対しても同程度の高速表現を求める事になろうと推測されるからである。これに対する方策の一つに文献[1], [2]に記した様に従来の一次元メモリを使いつつ、アドレス値の制御のみで行う方法があるが、メモリのアクセス順序が全体としてはランダムであるので一括アクセスが行ない難く、どうしても高速のメモリを必要とする。

補足の最後として第5図の総合処理部の問題に触れてみる。出来るだけ自由な画面構成を



望む立場からするとグラフィックディスプレイ装置といえども第5図に示された機能部分をすべて持つべきであるが、第一段階としては、まず基本的な機能から取りあげてゆく事にする。まず第一に必要となるものは(11)、(12)式を実行するハードウェアであるが、この場合領域規定情報間の演算は必要としないので、(11)、(12)式用に構成された属性情報処理部があればよく、各領域規定情報は直接ここへ供給される。但し(11)、(12)式における $a_z$ を集約属性、つまり走査途中では変動しないものとすれば、単位走査処理部に優先順位を与える事により、表示単位の下上関係の定義を、そのデータを設定すべき単位処理部の選択の問題に置き替える事が出来る。この場合は、領域規定情報処理部が(12)式の処理を受持ち、入力された領域規定情報の中で最も優位側のものを選択する操作を行うのである。

これまで述べて来た考察をもとに、いよいよ装置の具体化に移るのであるが、第一段階の目標としてまとめたのが第7図のシステムである。基本要素のみによる小規模な構成ではあるが、これでも当面の使用目的に対しては十分な性能を有すると考えられた。この時点では画像メモリ型の表示単位を特に必要としなかったなのでこの第一次案からは除かれている。又優先制御については、立体投影画の様な表示を当面必要としなかったので、前述のハードウェア固定順位方式を試みたいと考えた。単位走査処理部の出力は領域規定情報のみ、つまり



第7図 ディスプレイ装置の基本構成

各走査処理部当り 1 ビットの領域状態信号であり、色等の表示出力の値は中央制御部より表示出力部に与えられる。

表示単位の詳細はすべて境界点座標を表わすものであり、操作指令装置或いは外部システムより入力されるものである。表示単位を動的に制御するためには、これを実単位には関係のない単なるパターンとしての座標データ群と、その時々を表示意図によって定まる、表示画面上に占めるべき位置、大きさ、傾き等のデータとの組合わせで表わした方が便利である。とすれば第 7 図単位走査処理部内の表示単位原データメモリには前記組合わせの形態のデータを格納し、データ変換部で座標変換を施すと共にこれを境界線データに変換するのが適当であろう。しかし実際の装置設計の段階に至ると、この辺りの構成の仕方に種々の選択があり得る事が明らかになって来る。

例 1 第 7 図④の全範囲が一つの単位走査処理部であり、データ変換部も各処理部毎に備えている。

例 2 データ変換部を⑤の如く、幾つかの処理部で共有する。

例 3 単位走査処理部を⑥の範囲とし、データ変換部までを中央制御部の機能に含めてしまう。

例 1 から例 3 に移るほど処理能力は低下するが装置は簡単になる。

中央制御部については、当面通常型の計算機システムで充分と考えられ、今後構想の発展に伴って、必要に応じた専用ハードウェアを加える事でよいように思われた。又筆者の目的に対してのみの検討ではあるが、この方式では画面設計の負担が従来より軽くなると予想されるので、マイクロプロセッサ程度の中央制御部であっても、ホスト計算機を、動的表現のための変動パラメータの単なる供給源として位置づける事も可能な様に思えた。勿論中央制御部が強力であるほど、又ホスト計算機の寄与が大であるほどシステムの能力は拡大する訳であるが、最も小規模の構成であっても可成り独立性の強い装置が実現出来そうに思われたのであった。

## 6. 製作例

### 6.1 基本計画と機能確認モデルの製作

5.において説明した様に、昭和 51 年頃よりロケット管制での使用を念頭に置いての技術的検討が行われ、筆者なりに一応実現への目途はつけていた。その後ロケット管制の分野でディスプレイ装置を追加する計画はなかったが、駒場の振動・衝撃に関する環境試験システムにディスプレイ装置を備える計画があった。この装置に要望された性能は、本体の計算機のソフトウェアに余り負担をかけずに高速の図形表示を行える事であった。これは筆者の検討して来たシステムの特長に合致するものであった。ただし問題は開発予算が全く用意されていなかった事であり、実体は何も無い状態ではメーカへの製作依頼のしようもなく、そこで次の様な計画をたてた。まず全体計画として

- (1) 製作台数を 2 台とする。
- (2) 第一号を実用実験機として、実用にも使用し得ることを目標にして自作する。
- (3) 第一号機の動作が確認された時点で、第二号として実用機の製作をメーカに依頼する。

その仕様はその時点の第一号機のそれにはほぼ等しいものとする。

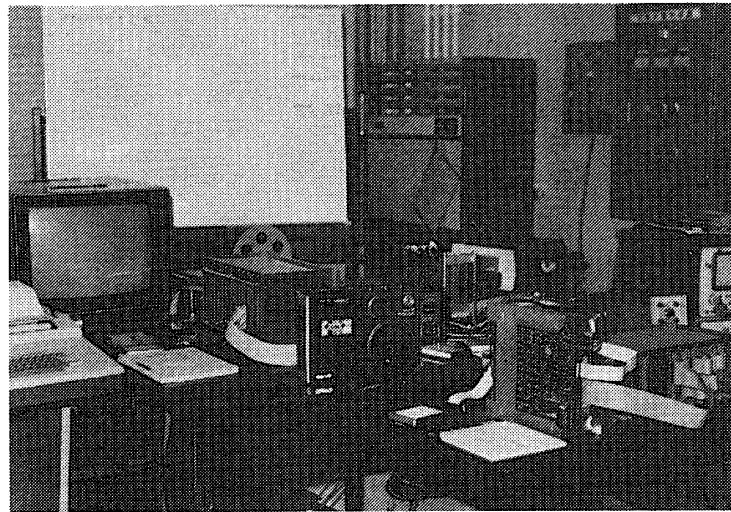
- (4) 実用機完成後は、実用実験機において性能向上、機能追加等の開発作業を進めてゆく。
- (5) 実用実験機で新たに得られた成果は機会をみて実用機に組み入れる。
- (6) 基本機能の動作が確認出来るまでを第1期とするならばこの第1期完成予定を第一号機で昭和54年度末、実用機で昭和55年度末とする。又元々ロケット管制システムの改良計画とも深い関係があったのでKSCにおける発射実験にも参加する事を計画する。

次に装置の仕様に関しては

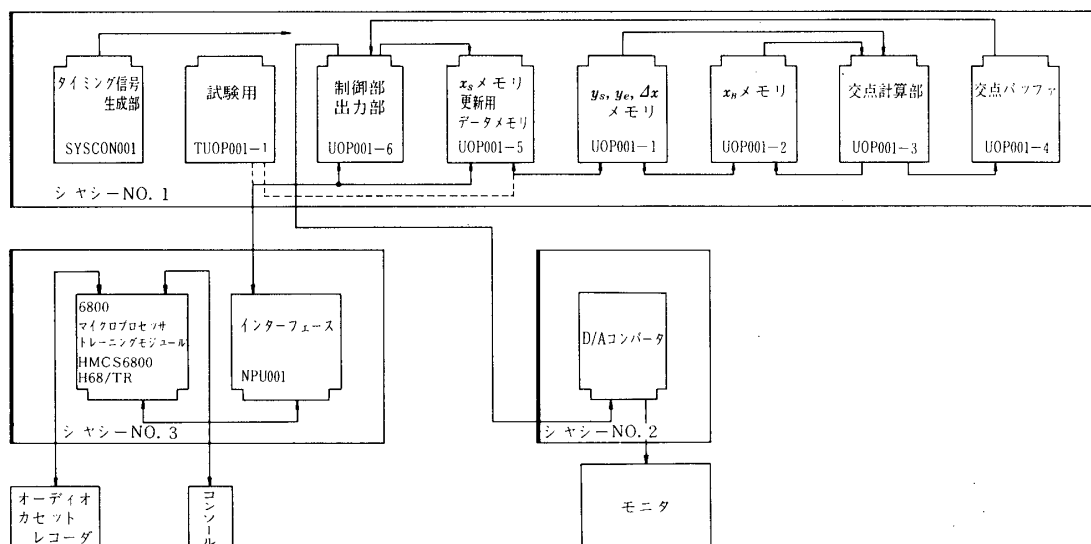
- (1) 画素数は $512 \times 512$ を目標とするが実用実験機では利用出来るモニタの関係で最高限度は $720 \times 512$ 、基本クロック30 MHzとした。一方自作であることから上記性能が出せない場合も有り得るので最低限度として $256 \times 256$ を設定した。つまり部分的に外注等を行ってでもこの最低線は達成しようと言う事である。
- (2) 上記に関連して実用実験機は30 MHz以下の如何なるクロックでも動作する様設計する。
- (3) 1表示単位の規模は、その境界点数で表わされるが、これは大雑把に20～100位を目標とし、詳細設計の段階で検討する事とした。
- (4) 3.に関連するが、単位走査処理部の数、つまり同時に走査可能な表示単位の数は20～50位を目標にする。1表示単位の点数が少くなる場合は一つの図形を複数の表示単位に分割する事になるので走査処理部の必要数が増加する。
- (5) 画面更新の速さについては、全表示単位の夫々において全境界点を移動させる最大規模の表示変更が1フレーム周期で実行出来る事を目標とする。但し開発過程ではこれより低い速度で使用しなければならない事も予想されるので、回路及びソフトウェアの設計上この点を考慮する。
- (6) 階調については、R, G, B各色8ビット、256階調のD/Aコンバータ基板が入手出来たので、最初からこの仕様で設計する事とした。
- (7) 走査処理部の機能に出来る限り後述の「線表示」を加えることとしたい。これにより同一のパターンデータのまま、内部ベタ塗り、輪郭線表示を簡単に切り換える事が出来る。
- (8) 中央制御部は当面モトローラ6800系マイクロプロセッサシステムを利用するが、後日レベルアップが容易に行なえる様インターフェース等を考慮しておく。
- (9) 前記環境試験システムはパナファコムU3002台で構成されているが、ソフトウェア上表示制御に時間を割く余裕がないが、共通バスには余裕があると推定された。そこでICUの標準構成品であるDMOユニット[3]をU300側に設け、ディスプレイ装置側はこれとの間にインターフェースを設けて、この系でデータを取得する事とした。尚KSCのロケット管制システムとの接続も同一の方式とする。

が方針とされた。これに従い設計・製作にとりかかる訳であるが、原理的に不明確な事項は残っていなかったので計画実現に対して特に不安は無かったものの、何分筆者にはこの種の装置の自作経験が皆無であったので、製作練習を主目的として小規模のモデルを製作するこ

ととした。単位走査処理部は本システム独特の部分であり、事前に機能を確認しておく方がよいので、これをモデルの対象とすることとした。このモデルは昭和 52 年 9 月頃より製作をはじめ、次の実用実験機の設計時期と重なっているが 53 年 7 月には全実験を終了している。第 8 図に実験中の状態を、第 9 図に概略の構成を示す。図中完成購入品は、モニタ、 $D/A$  コンバータ、日立製 6800 マイクロコンピュータトレーニングモジュール [4] 一式であ



第 8 図 実験中のモデル機

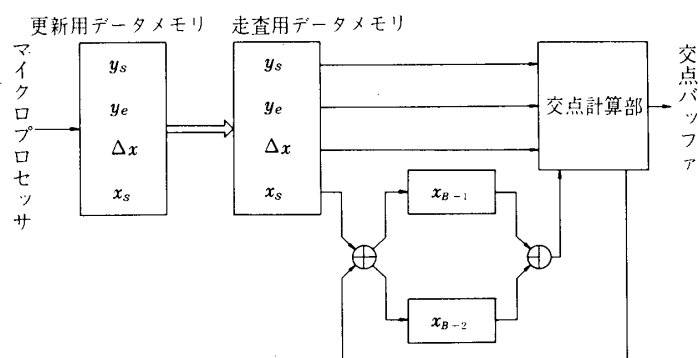


第 9 図 モデル機の構成

る。他の基板はすべてラッピング手配線によった。

第9図シャシーNO. 1の8枚の基板のうちUOP 001-1~6の6枚が第4図の方式による1ヶの走査処理部を形作っている。これは第7図の⑥部に相当し、単位走査処理部の他の機能は6800マイクロプロセッサシステムが受持つ。SYSCON 001は基本タイミング信号の生成用基板である。NPU 001は6800システムと走査処理部との間のインターフェースであり、シャシーNO. 3内にあって6800基板とは直接バスで結ばれている。尚シャシーNO. 1内のTUOP 001-1はテスト専用の基板である。これはNPU 001と結ばれ、走査処理部内の主要制御信号の夫々を個別にマイクロプロセッサで制御するためのものであり、UOP 001-1~6基板内の回路はこれに対応出来る様設計されている。このためIC数は可成り多くなっている。シャシーNO. 2にはD/Aコンバータのみが収められている。尚総合処理部は製作されなかったで、その分の機能をUOP 001-6の中に設け、ここから最終のR, G, B輝度値をD/Aコンバータに対して送り出している。

次に第4図の回路を実現する際に生じた問題について述べるが、このモデル機の回路には第4図と異なる点があり第10図にそれを示した。グラフィックディスプレイ装置では、表示の視認性等からその走査周期が規定されてくる。画面更新のタイミングも当然この走査周期に左右される。走査用データメモリの内容は境界線のデータである。従ってこの1ビットの変更が表示画面に及ぼす影響は大であり、場合によっては全画素が影響を受ける。このためデータメモリの書替えを部分的に行う事は避けるべきであり、このためフレームとフレームの間の垂直帰線期間に行わねばならない。検討の結果、第10図の如く、ほとんど同じ構成のメモリブロックを前置させ、これを更新用データメモリとして中央制御部からのデータ更新は、これに対して行われる様にした。更新されたデータはフレーム間に一括して走査用データメモリへ転送される。 $x_B$ メモリの構成法については種々の案が考えられたが、このモデル機では10図の如く通常のRAMによるメモリブロックを二面設け、これを読み出しブロック、書き込みブロックと切換えて使用する事とした。 $x_B$ データは走査の結果、変化して



第10図 モデル機のメモリ配置

しまうので、走査に先立ちフレーム間に  $x_s$  ブロックよりどちらか一方のブロックへ一括して転送される。これは更新の有無にかかわらない。この初期ロードされた側のブロックが最初に読み出し側に設定され、交点計算部を経て他方のブロックに至り、書き込みが行われる。次の水平走査帰線期に両ブロックが切り換えられる。以下これを繰り返す。

次に使用するメモリの選択について述べる。一水平走査周期を  $T_H$  とし、メモリのリードサイクルタイムを  $t_{cr}$  とする。ここでそのメモリについて第 4 図又は第 10 図の方式で一回の水平走査期に扱う事の出来るデータ数の最高限度を  $N_{dx}$  とすると

$$N_{dx} = T_H / t_{cr} \quad (19)$$

一チップに格納したいデータの数  $N_d$  とした時に

$$n + 1 > N_{dx} / N_d \geq \text{整数 } n \geq 1 \quad (20)$$

の関係が成り立つ  $n$  が存在しなければならないが、この場合 1 データは  $n$  回に分割して読み出される事となる。実際に使用されるメモリは  $n \times N_d$  アドレス分となる。メモリ容量が  $N_c$  アドレス分であるとき

$$\mu = (n \times N_d) / N_c \quad (21)$$

はメモリの利用効率を示している。本モデル機では短期間に入手可能なものの中から選択する必要があり、結局富士通 MB 8112 E [5] (インテル 2112 A-2 相当品, 256 × 4 ビット構成,  $t_{cr} = 250$  ns) を用いる事となった。一方各種タイミング信号は基本クロックから作り出す事としていたのでメモリのリードサイクルタイムも基本クロック周期の整数倍となる。基本クロック 30 MHz とすると結局  $t_{cr} = 266.7$  ns となった。  $N_{dx} = 31800 / 266.7 \div 119$  となるが、若干余裕をとる必要があり、112 とする事とした。  $N_d$  については 112 でも 56 でも良かったが結局 112 とした。  $n = 1$  である。メモリ利用効率は  $\mu = 1 \times 112 / 256 \div 0.44$  となり低率である。

次に交点計算部の前後に、つまり  $x_B$  メモリブロックの前後にレジスタを設け、この部分の処理をパイプライン状とした。従って 1 データ分の処理を 266.7 ns 以内に行う様計画した。これにはショットキー型の TTL を使う事により充分余裕がとれたので、ここに境界線表示の機能を加える事とした。今まで第 4 図の回路を、1 境界線データについて最大 1 ケの交点に関する記録を交点バッファに書き込むものとして説明して来た。この方式のままで線を書く場合は、 $y_s, y_e, \Delta x$  が等しく  $x_s$  のみ或る値離れた 2 データが必要であったが、もし同じサイクル内に垂直の場合は 1、傾きのある場合は  $\Delta x$  隔った所にもう一つ交点があるとした処理が可能であれば、同一形状の図形を同一データ数で、面と輪郭との何れの表示も出来る事となる。この場合第 4 図の交点バッファのアドレス入力の接続先を加算器の入力側から出力側へ移す事と、加算器周辺に若干の追加回路が必要となる。

交点バッファとしては基本クロックの 1 周期つまり管面の 1 ドット分の表示時間に等しいサイクルタイムのメモリが必要である。これは読み出し時の動作から定められる条件である。ここではバイポーラ RAM, フェアチャイルド 93425 A [6] (1024 × 1 ビット, アクセス

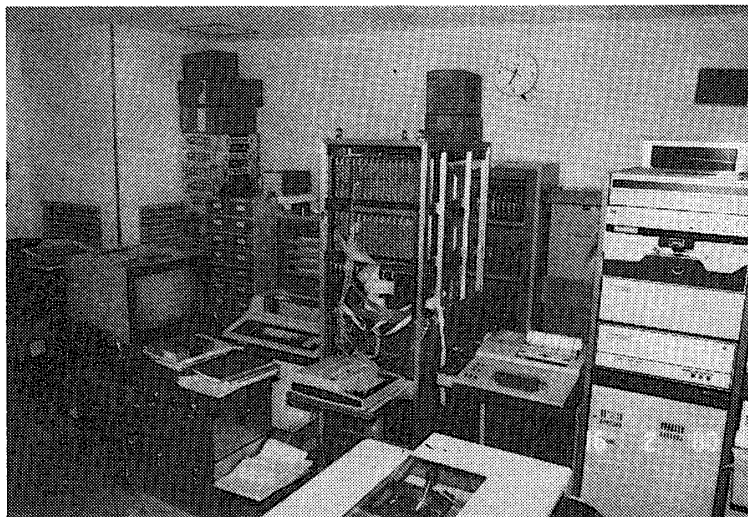
タイム Max 30 ns) を採用した。当時これと同等性能の MOS RAM インテル 2125 H は発表されていなかったが、同社 2125 A も実験的に使用した。アクセスタイム Max で 45 ns であり、基本クロック周期 33.3 ns に対して 100% の確実性は保証出来ない訳であるが、筆者の購入したものについては、充分代用が可能であった。尚この交点バッファのクリアだけのために 1 水平周期分の時間を割り当てる事としたので結局 3 面バッファの切替制御方式となった。

データについては、 $y_s$ ,  $y_e$  に夫々 9 ビット、 $\Delta x$  に 19 ビットをあてた。 $x_s$  は 10 ビットである。 $\Delta x$  の上位 10 ビットが  $x_s$  の部分に対応し、下位 9 ビットは小数部である。 $x_e$  もこの小数部を持つのでやはり 19 ビット必要である。実際のものは他の理由もあって 21 ビットであった。当然の事ではあるが、 $x_e$  のうち交点バッファのアドレスとなるのは、その整数部であるところの上位 10 ビットである。尚この他に線表示を示すための 1 ビットがあり、このため前述の線表示と面表示のデータが 1 表示単位中に混在出来るが、これは図形データ管理上、有利か否か、現在のところ結論は出していない。

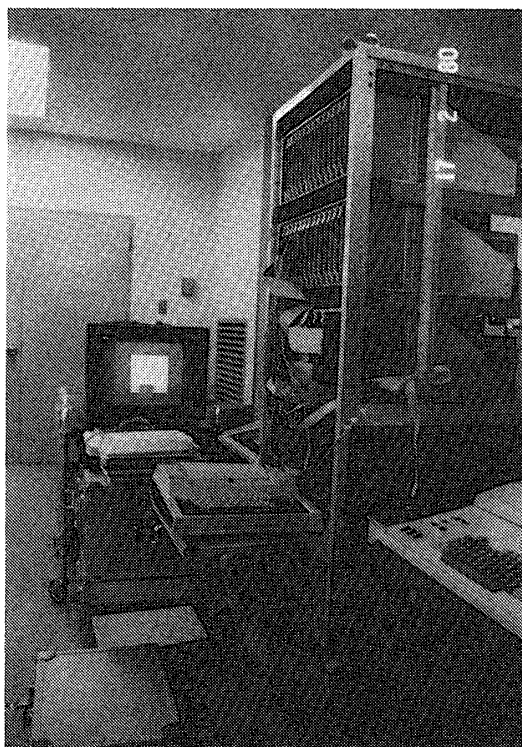
## 6.2 実用実験機の製作

モデル機の経験を生かしつつ、昭和 53 年度初めより実用実験機の設計を進めた。この設計と並行して製作にも着手し、昭和 53 年度末には単位走査処理部をほぼ完成させ、仮の出力制御部を用いて画像の表示が可能となった。表示画面はモデル機と同様 720 × 512、基本クロックは 30 MHz であった。以上の成果が得られたので、昭和 54 年度早々、株式会社グラフィカに実用機の製作を依頼した。実用実験機はその後開発作業を進め、同年中に優先制御部を組み込み、又座標変換等を行うデータ変換部を 4 ビットスライス型マイクロプロセッサによるものに改め、表示速度を向上させた。この段階で充分実用に使える性能を得たと判断したので、昭和 55 年 2 月 M-3 S-1 号機の発射時に、KSC の保安管制用計算機システムに接続し、実用実験を行う事とし、実際に 3 箇所のレーダステーションの追跡状況の相関を示す画面を表示し、その特性を確認する事が出来た。紙数の都合もあり、本機の詳細な報告は後の機会に譲る事とし、以下にはその概要を記す事にする。第 11 図、第 12 図は KSC 計算機室内における実用実験機である。又全体の構成を第 13 図に示す。

以下、第 13 図を中心として、主としてモデル機において述べた事項と関連する部分について説明する。データメモリについては、計画当時 8112 E が入手困難 となってしまったので、インテル 2125 A [7] 及び同一仕様の富士通 8125 Y を採用した。本機はモデル機同様ほとんどラッピング配線基板による実装方法をとることとして居り、この条件のもとで出来る限りコンパクトにまとめる事も目標の一つであった。これは遠隔地での実験も念頭にあり、又実用機との関連もあったからである。一方ソフトウェアも含んでの総合実験のためには、可能な限り多くの表示単位を扱いたかった。以上の点を勘案した結果、扱う表示単位の数、つまり単位走査部の数を取りあえず 32 とし、第 13 図シャーシ 00、シャーシ 01 の同構成のシャーシ 2 台に収める事とした。尚これは拡張可能である。ここで 1 シャーシ当り 16 の表示単位を扱う事となった訳であるが、種々検討の結果、走査用データメモリについてはこの 16 表示単位分のデータのすべてを 1 枚の基板に集中して格納する事になった。(UOP

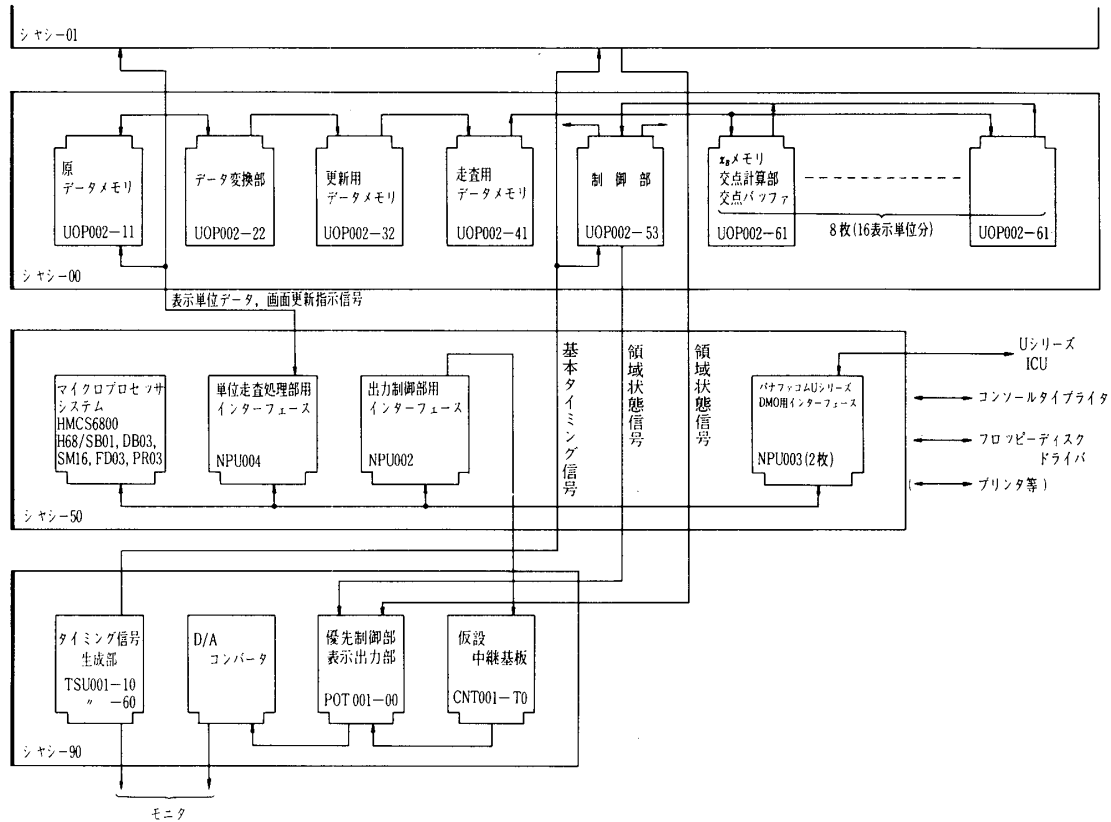


第 11 図 Uシリーズと接続した実用実験機



第 12 図 動作中の実用実験機





第 13 図 実用実験機の構成

002-41)。これは第7図において⑨の様にまとめた事になる。更新用データメモリも同様の構成とした。メモリ中のデータはアドレス順に「表示単位NO.0第0データ，表示単位NO.1第0データ，---，表示単位NO.15第0データ，表示単位NO.0第1データ，---」と格納されている。これを順次読み出してデータバスに乗せる。一方 $x_B$ メモリ，交点計算部は2表示単位分で基板1枚（UOP002-61），合計8枚にまとめられているが，これらが夫々異なるタイミングで自分の分のデータをデータバスから拾い上げる方式とした。 $t_{cr} = 33.3 \text{ ns}$  とすると(19)式から  $N_{dx} = 31800 / 33.3 = 954$  これを16単位で割ると約59となり，最終的には1表示単位当り56データと定めた。 $N_d = 56$ ， $n = 16$  であるので(21)式より  $\mu = 16 \times 56 / 1024 = 0.875$  となりメモリ利用効率は大幅に改善された。但しその反面，制御を複雑にする等のマイナス面もあり，今後の課題の一つであろう。又実際のメモリのリードサイクルタイムが，この目標値33.3 nsより長くなるほど1表示単位当りのデータ数が減少し，利用効率も低下するが，本機では入手したメモリのうちサイクルタイムの早いものを選んで，この走査用データメモリ基板に搭載し，最終的に  $t_{cr} = 33.3 \text{ ns}$  を実現した。一方残余の遅いメモリは更新用データメモリに用いられた。

$x_B$  データ用メモリはバイポーラ FIFO (First-In First-Out) 型メモリとし MMI 67401 [8] ( $64 \times 4$  ビット) を採用した。この場合  $x_B$  データは FIFO～交点計算部～FIFOと循環

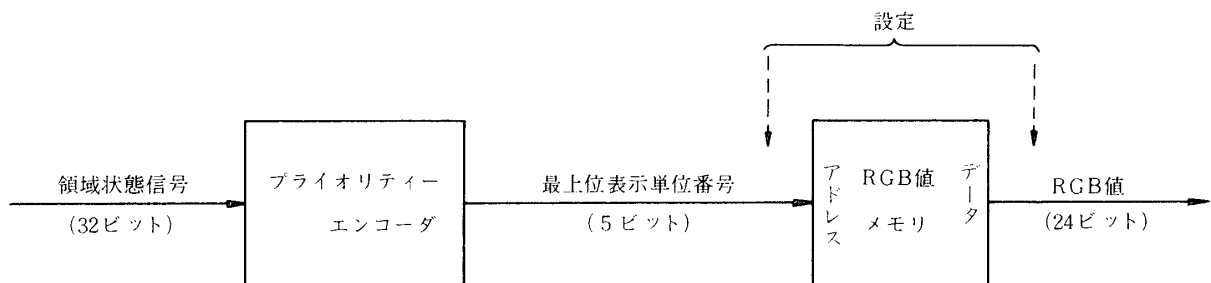
させる事が出来るので一組でよい事になる。走査処理部のその他の部分については、概ねモデル機での思想が引き継がれている。又交点バッファは走査用データメモリと同様 2125 A の早いものを用いた。

基板 UOP 002-11 の原データメモリ (日立 HM 472114 AP-2) は 16 表示単位分のデータを境界点の  $x, y$  座標及び座標変換パラメータの組として格納している。これを表示単位データ (図形データ) の管理の面で眺めると、中央制御部の制御下にある一群のデータファイルの一部を構成しているが、座標変換、境界線データへの変換に際しては、中央制御部から切り放され、シャシー毎に固有のデータ変換部の制御下に入る。

データ変換部の基板 UOP 002-22 は第 7 図©型の設計であり、16 単位分の処理を実行するものである。これには 4 ビットスライス型マイクロプロセッサ AMD 2903 [9] を 4 チップ、16 ビット構成で使用した。マイクロプログラムは 54 ビット巾であり、マイクロプログラムメモリにはバイポーラ PROM MMI 6349 (512 × 8 ビット) を 7 チップ使用した。

タイミング信号生成部と出力関係はシャシー 90 に収められている。優先制御については、とりあえず不透過性表示のみとして設計した。この基板 (POT 001-00) の設計概念を第 14 図に示す。プライオリティーエンコードからは同時に入力された信号のうち、最上位の H 信号を示す様エンコードされた信号が出力されるので、これをアドレス値として、メモリに予め設定されていた RGB 値を読み出して  $D/A$  コンバータに送る。この方式ではプライオリティーエンコード入力端子と各单位走査処理部とのハードウェア上の接続関係で表示単位の上下順が定まる。従って表示単位の上下制御は、中央制御部のデータ～走査処理部の対応づけ操作の問題に帰着する。尚本基板には、領域状態信号がすべて L の場合に別途設定されていた背景色レジスタの内容を出力させる機能、ネガポジ反転操作機能、表示単位出力個別抑制機能が付加されている。

以上の各部を制御する部分をまとめたのがシャシー 50 であり、中央制御部に相当する。これは日立 6800 マイクロプロセッサ SBC シリーズ [10] の各種基板を中心として構成されている。単位走査処理部、表示出力部、パナファコム U シリーズ DMO 装置に対しては夫々インターフェース基板を設計した。



第 14 図 実用実験機における優先表示制御方式

以上 4 台のシャーシ及びフロッピーディスクドライバを標準型ラックに装着し、ラック外の装置としては、コンソールタイプライタ（カシオ製タイピュータ又は TI 製サイレント 700）とモニタがある。プリンタ等その他の装置を必要に応じて接続する事があるが、常設ではない。

第 13 図に記入された各基板及び開発過程でのみ用いられたその他の基板（約 12 種）のうち 6800 SBC シリーズの基板と  $D/A$  コンバータ基板のみが完成購入品であり、その他については、第一枚目を自作し、複数枚製作する場合は、一枚目のテスト終了後第二枚目以降のラッピング作業を外注すると言うのが標準の形態であった。

KSC において実験的に使用した 55 年 2 月の時点では、中央制御部のソフトウェアは未だ整備されておらず、仮のプログラムで稼働させた。確実を期して監視用画面を 16 表示単位以下で構成する様計画し、走査処理部シャーシ 1 台を予備・待機状態で実験に臨んだ。DMO 系統によるデータの取得、これによる画面の制御共に予定通り実行され、この構成のハードウェアのままで、ソフトウェアを整備する事により、充分実用目的に使用し得るとの確信を強くした。

その後、本機については各種の改良及び機能追加の計画を進めると共に、中央制御部のソフトウェアの開発が進められている。

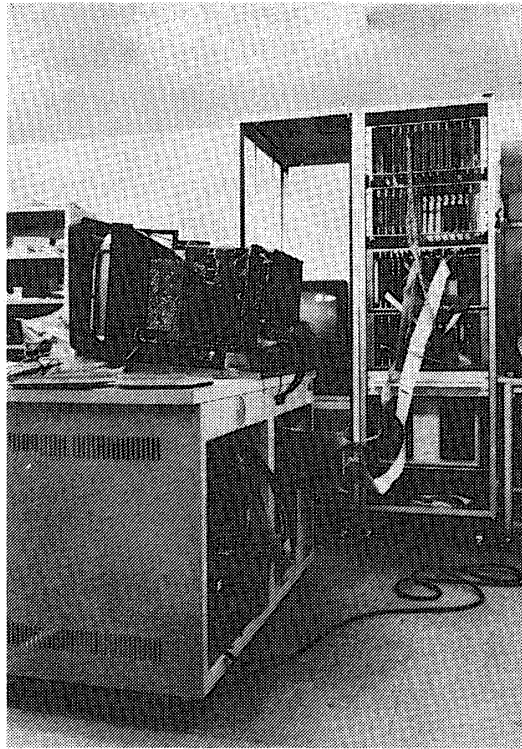
### 6.3 実用機の製作

既定方針に従い、昭和 54 年に実用機をグラフィカ社に発注した。本機の仕様は 6.2 で説明した実用実験機のものにはほぼ等しいものである。第 15, 16 図は本機の写真である。実用実験機では、4 台のシャーシのうちシャーシ 50 と 90 に将来の追加基板用の未実装部が残されているが、本機では一応全体を 3 台のシャーシにまとめてある。ただし、装置を運用に供するためのソフトウェアは昭和 55 年度中に筆者が作成する事になっており、このため、これに関連した中央制御部機能の一部を除いた状態で製作された。従って当初は実用実験機と接続して運転する等、仮の運用形態をとる事もあり得るが、ソフトウェア関連部分の整備が完了した時点で、前述の環境試験システムに組み込まれる予定である。

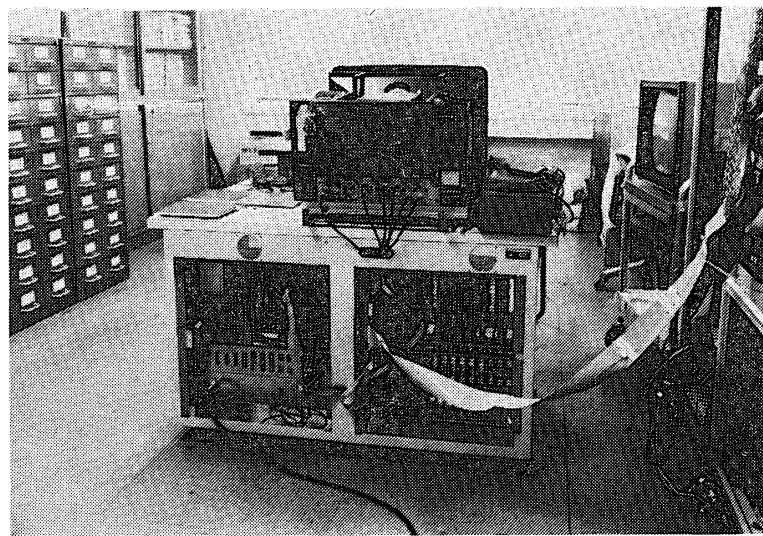
## 7. あとがき

元来この領域情報の問題は、筆者にとって全く専門外の分野であった。10 年程前に計算機での計算結果を大量に COM 等で図化していた時期があったが、その頃の仕事が、この種の問題へ関心を持つ発端であったかもしれない。前半に述べた領域情報処理に関する基本的な考え方は、その頃の考えからそれ程進歩していない様に思える。関連する研究の調査も含めて、未整理、未検討の部分が多いまま、当面の必要に迫られて、後半に述べた様な非常に特殊化された応用分野につき進んだと言うのが真の姿であろう。

本文に述べた基本的な処理法が、すべての領域情報の処理に適用出来るのであれば、例えば画像処理の分野への応用なども考察してみるべきであろう。残されている問題は大変多いと考えている。



第15図 実用実験機に接続して試験中の実用機(1)



第16図 実用実験機に接続して試験中の実用機(2)

筆者一人の力でどれ程の進展がみられるか全く疑問であるが、とにかく実用実験機に順次機能を付加してゆきつつ、実証部分を増やしてゆくのが当面の課題と考えている。

1980年11月27日 新設部(工学)

## 参 考 文 献

- [ 1 ] 中野 旭；領域情報の処理方式，特許出願公開昭 54-7241，昭和 54 年 1 月 19 日
- [ 2 ] 中野 旭；2 次元領域情報の処理方式，特許出願公開昭 56 - 11558，昭和 56 年 2 月 4 日
- [ 3 ] 富士通；PANAFACOM U シリーズ ハードウェア ICU 仕様編 2 版，昭和 53 年，54
- [ 4 ] 日立；トレーニングモジュール H68/TR ハードウェアマニュアル，昭和 52 年 6 月
- [ 5 ] Intel: Data Catalog 1977, 2-72
- [ 6 ] Fairchild: Bipolar Memory Data Book, 1976, 7-93.
- [ 7 ] Intel: Data Catalog 1977, 2-86.
- [ 8 ] MMI: Bipolar LSI Data Book, 1978, 5-48.
- [ 9 ] AMD: The Am2900 Family Data Book, 1979, 2-31.
- [10] 日立；シングルボードコンピュータ H68SB01-1 ユーザーズマニュアル，昭和 53 年