The University of Osaka
Institutional Knowledge Archive

Title	DAPシミュレーターの開発と応用
Author(s)	吉田, 勝行; 田中, 茂
Citation	大阪大学大型計算機センターニュース. 1976, 23, p. 29-63
Version Type	VoR
URL	https://hdl.handle.net/11094/65328
rights	
Note	

The University of Osaka Institutional Knowledge Archive : OUKA

https://ir.library.osaka-u.ac.jp/

The University of Osaka

DAP シミュレーターの開発と応用

（同一のプログラムで、プロッターとラインプリンターの両方に同一の図を描かせるためのシステムの設計）

大阪大学教養部　吉田勝行

大阪大学基礎工学研究科　田中　茂

阪大大型計算機センターの自動製図装置ドラフターとＤＡＰシステムによる図が美しいことは万人の認めるところですが，研究者にとって使用不便であることもまた万人の認めるところです。これはドラフターが，カルコンプ系の機械のように，計算結果をすべて図化して出力するという設計思想で開発されたものでないため仕方がないともいえますが，そのためか使用に拒絶反応をおこす人をまま見受けます。

以前筆者がプログラム相談員をしていた頃相談に見えた医学部のさる方は，Ｂジョブで３００枚のプリント用紙に端から端までびっしりとプリントされた数値を見せながら，

「数値を計算機でこしらえるのは１５分足らずやけど，一枚一枚手でグラフにするのに半年はかかるやろな。」

とげっそりしていましたが，いくらすすめてもドラフターを使うつもりはないようでした。むろん新しく図をかくプログラムを組んでディバックをくりかえしているうちに，半年ぐらいすぐたってしまうので，時間がもったいないというわけです。

もう少し違った拒絶反応のあらわれ方もあります。先日センターの受付でガミガミやっている人がおりました。

「このプリント用紙にかけてくる図をそのまま論文に使おうと思っとったのに，ラインプリンターの管理が悪いせいで，一部がかすれてしもて使われへん。なんとかしてくれ。」

機械で図をかこうとは思うが，ドラフターで清書までしようとは思わないというわけです。

こうした例や筆者ら自身の経験からみて，計算結果の図化システムを研究の道具として使おうとする場合，そのシステムは次のような条件をみたす必要があるようです。

a．プログラムを組み始めてから目的の図が手に入るまでの実質的なターンアラウンドタイムが短いこと。特に機械によるディバッキングのターンアラウンドが短いこと。

b．図形やグラフは概形を知りたいことが多いので，とりあえずかけてくる図は，ラインプリンターによる図程度の精度でよい。それらの図のうちで本当に清書の必要なものだけを精度を上げて製図出来ること。

したがって，阪大大型計算機センターの自動製図システムＤＡＰを基礎にしてこれらをみたすには，次のような設計条件をみたすシステムをあらたに開発すればよいことになります。

(1) ＤＡＰ用に作られたプログラムに，当該システムのカードデックをつけたせば，もとのプログラムになんら手を加えなくても，ラインプリンターによる図が出力される。

(2) 当該システムをつけ足して通ったプログラムは，当該システムをはずしても必ず通り，ドラフターによる図が得られる。

(3) 当該システムによりグラフ等をラインプリンターで大量に描いた場合，そのうちの任意のものを手軽にドラフターで清書出来る。

(4) 当該システムを使った場合も，オープンバッチジョブやＡジョブ等の記憶容量やＣＰＵ時間，外部記憶操置の使用が制限されたジョブに通せる。

これは結局ＤＡＰによる描画を，ラインプリンターでシミュレートするシステムを作り出すことになります。それゆえ，以下ここに述べる当該システムを，ＤＡＰシミュレーターと呼ぶことにします。

むろん上に述べた４つの設計条件をすべて満たすシステムを作り出すのは不可能です。そこで(4)の「オープンジョブに通せること」という条件を最優先条件とし，以下のように要求条件を制限することにします。

(i) ＤＡＰシミュレーターには，ＤＡＰシステムのサブルーチンのうち，ＬＩＮＥ１，ＡＲＣ１等の図形を処理する上で基本的な機能を持つサブルーチーンのみを採用する。そのかわり，採用していないサブルーチンが利用者のプログラムでＣＡＬＬされた場合には，無視したむねのコメントを出力し，注意を喚起する。

(ii) 利用者のプログラムが，ドラフターで大きな図をかくことを予定して作られている場合でも，ＤＡＰシミュレーターでは，ラインプリンターによる出力は，一度の実行で，プリント用紙一頁分の大きさの，図の一部分のみがプリントされるものとする。ただし図全体の任意の場所を一頁分出力出来るようにする。

(iii) 設計条件(2)は，ＤＡＰシステム自体に不明な点や不十分[＊1]だと思われる点がいくつかあり，またラインプリンターを用いる場合の限界もあって完全には実現し難いが，出来るだけ完全に近づくよう配慮する。

このような条件を満たすシステムを開発するについて，以下このＤＡＰシュミレーターシステムを２つの部分に分けて考えます。その部分の名称と機能は，次のとおりです。

(A) ＤＳ－１：ラインプリンターで図を描く場合に，画面の配列をプリントアウトしたり，ドラフターで清書する場合の図の位置を制御するサブルーチンの集合。

(B) ＤＳ－２：ＤＡＰシステムの図形処理サブルーチンをラインプリンター向きに処理するサブルーチンの集合。

＊１　付録参照

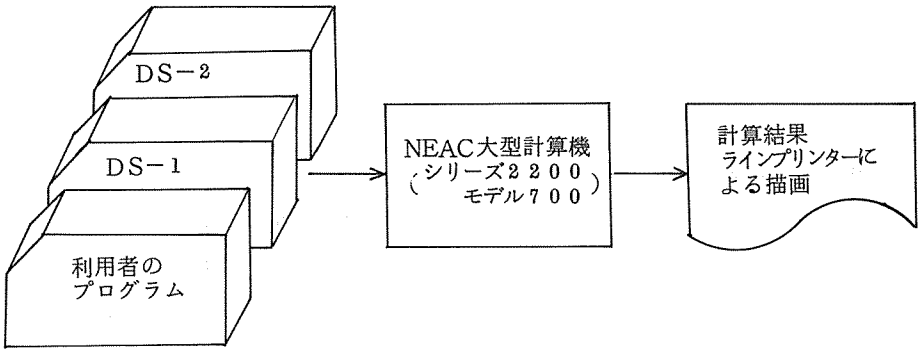


図 1.　DAP シミュレーターを用いて、ラインプリンターで描画を行なうときの
処理過程の概略

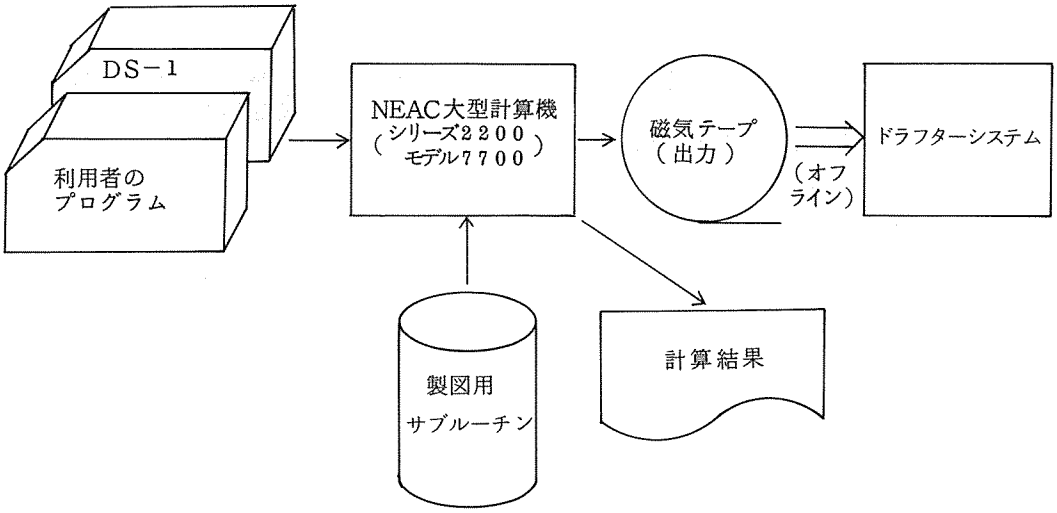


図 2.　ドラフターで描画（DAP シミュレーターの一部を用いている場合）する際の
処理過程の概略
（DAP シミュレーターのサブルーチンを引用していないときは、DS－1の
カードデックをとり除いてもよい。）

図１および図２に示すように，こうしたＤＡＰシミュレーターの一部ないしは全部を利用者のプログラムにつけ加えて通すことにより，手軽にラインプリンターでディパックが出来たり，ドラフターで清書が出来るわけです。

なお，このようなシミュレートシステムは，文献（２）にもプログラム例が報告されていますが，外部記憶装置を使用している，線分を描くアルゴリズムがプロッター用のものをそのま

ま利用している，サブルーチン構成がＤＡＰと異なる等，筆者等の意図しているものと随分異なっていて，あまり参考にはなりません。

以下では，筆者らの開発したＤＡＰシミュレーターシステムと，これを開発し使用した時点で気づいたことを中心に述べることにします。

1. ラインプリンターによる描画とドラフターによる描画の相違

ラインプリンターのプリント用紙１ページには，縦６０個，横１３２個の活字を打つことができますが，筆者らは，多少の余裕をみて，このうち，縦５６個，横１３０個を使用しています。すなわち，ラインプリンターによる描画は，図３のように５６×１３０のます目をもつ配列を画面として用意し，この画面内に活字を並べて図とするわけです。活字の大きさは，縦が１／６インチ＊1，横が１／１０インチで，図に示すます目以外に活字を置くことは許されませんから，ドラフターで描く図にくらべ，精度はあらく，解像力もよくありません。しかし図をかく速度は，ドラフターにくらべてはるかに速いという特長があります。

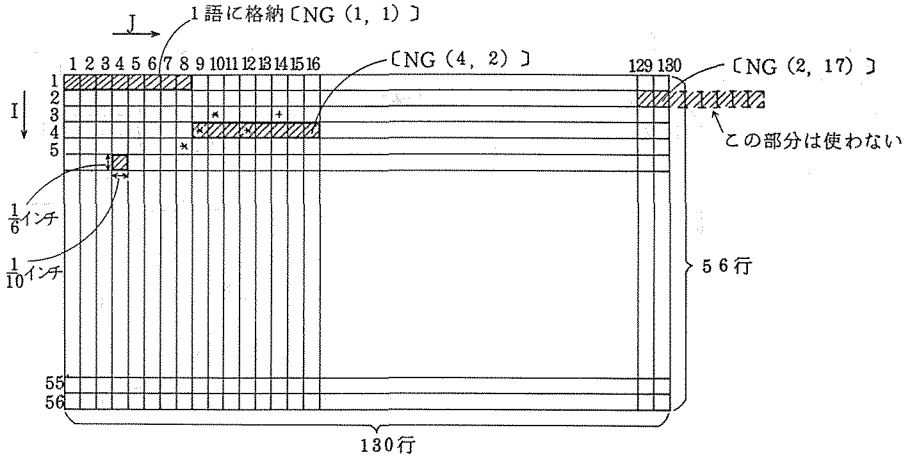


図 3. ラインプリンター描画を行なうための画面

（画面のます目は５６×１３０であるが、図のように８個のます目を１語としてあつかうので、記憶装置としてはNG（５６，１７）でよい。）

本システムでは，１つのます目には，１つの活字しか置けないという制限のもとで，図をかくようにしてあります。プログラムを工夫すれば，１ますに多重に活字を重ねて打つことも可能ですが，配列をいくつも用意する必要があり，記憶容量，計算時間が増加するうえ，見にくくなり，ラインプリンターのリボン等もいためますので，デザイン等の特殊な用途以外には，あまり実用的でないからです。

ラインプリンターによる描画とドラフターによる描画との主な相違点は，表１のとおりです。

＊1　１インチ＝２５.４㎜

表1. ドラフターによる描画とラインプリンターによる描画の主な相違点

描画出力装置の種類	ドラフター	ラインプリンター
線の描画	サブルーチンがコールされるごとに描く。（逐次的）	配列内へ一時，記憶しておき，まとめて出力を行なう。（一括的）
描画範囲	1200㎜×1000㎜ （有効描画範囲） 模造紙大として 1100㎜×790㎜ 清書用紙 1200㎜×900㎜	約330㎜×237㎜ （プリンタ用紙1ページ （活字56×130）分）
基本設定単位（精度）	0.02㎜/パルス （目で見てはぼアナログ的な図とみなしうる。）	縦1/6 inch（約4.23㎜） 横1/10 inch（2.54㎜） （目でみたとき，ディジタル的な感じがありありとする。）

なお以下では，画面という言葉は，ラインプリンターではプリント用紙1頁分の56×130のます目を，ドラフターではドラフター描画用紙1枚分をそれぞれ意味することにします。そして，図3に示すような画面に相当する配列NG（56，17）は，画面の配列と呼ぶことにします。

2. 座標系と原点

座標系は描画座標系（現座標系）を除き，図4のように，X軸，Y軸をとり，㎜単位で設定

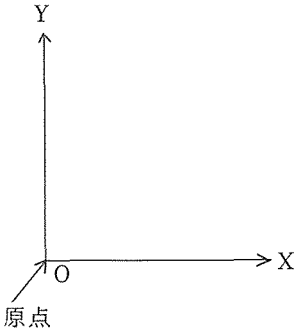


図4. 座標系

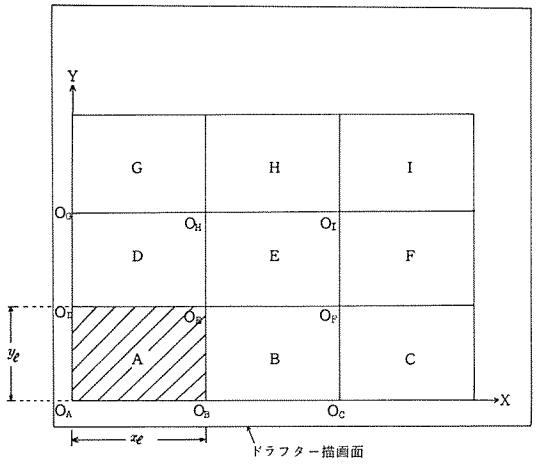
される座標系を想定しています。以下に当システムで用いる様々な座標系を定義しますが，ドラフター操作卓によるX MIRRŌR，Y MIRRŌR，XY MIRRŌR等の各キーは，一般的でないので当システムでは用いないものとします。

マシン座標系とマシン・ゼロ：これは，ドラフターに固定された座標系および原点のことで

す。マシン・ゼロは，ドラフター描画面の左下隅と定められています。

ドラフター座標系と初期原点（機械的原点）：ドラフターにスイッチを入れたとき，またはスイッチを入れたのち，マニュアルで原点移動させたとき，形成される座標系と原点です。すなわち，描画の際，まず描かれるダップスター・ナンバーＮ$\bar{O}$．のＮの文字の左下隅の位置が初期原点です。文献（４）では，これを機械的原点と称しています。

区画座標系と区画原点：ドラフターで後節において説明するサブルーチンＰＲＩＮＴ（１）を利用して描画するとき，ドラフター描画面をいくつかの区画に分割しますが，この区画の原点が区画原点です。また，この区画原点を原点とする座標系が区画座標系ですが，くわしくは図５を御覧下さい。



全区画数 / 図中の記号	4	6	9
A	第１区画	第１区画	第１区画
B	2	2	2
C		3	3
D	3	4	4
E	4	5	5
F		6	6
G			7
H			8
I			9

XO_AY：ドラフター座標系（O_A：初期原点）

O_A：第Ａ区画原点（いずれの場合も第１区画原点）、他についても同様

x_ℓ, y_ℓ：区画の大きさ（隣り合う区画原点間の距離）で、初期値は、３５０.０，２５０.０でサブルーチン$\bar{O}\bar{O}$KISAで変更可能である

図５．　区画座標系と区画原点

プリンター座標系とプリンター原点： ラインプリンターによる描画の際の画面の左下隅に位置するます目の中心を原点（これをプリンター原点という）とする座標系がプリンター座標系です。

描画座標系（現座標系）と描画原点（現原点）： 描画している際，現に設定されている現在の座標系が描画座標系で，この座標系の原点が描画原点です。

仮想座標系と仮想原点： あらかじめ自分が描く図を頭の中に想定した場合，その図に必然的に付随するないしは頭の中に必然的に想定する座標系と原点が，仮想座標系と仮想原点です。

プリンター基準点： ドラフター座標系上でプリンター原点と一致する点がプリンター基準点です。サブルーチンＫＩＺＹＵＮを使えば変更出来ます。

各座標間，および各原点間の関係は次の通りです。

仮想座標系＝ドラフター座標系＝第１区画座標系

仮想原点＝初期原点（機械的原点）＝第１区画原点

プリンター原点＝プリンター基準点

ドラフター操作卓の操作により，ペンの位置をマシン・ゼロに復帰させたときは，

マシン座標系＝仮想座標系＝ドラフター座標系＝第１区画座標系

マシン・ゼロ＝仮想原点＝初期原点（機械的原点）＝第１区画原点

サブルーチンＫＩＺＹＵＮ等により，プリンター基準点が変換されないときは，

仮想座標系＝プリンター座標系

仮想原点＝プリンター原点

座標軸変換ルーチン等により，座標系が交換されないときは，

仮想座標系＝描画座標系（現座標系）

仮想原点＝描画原点（現原点）

3. ＤＳ－１：描画制御用補助ルーチン

ＤＳ－１は，ＰＲＩＮＴ，ŌŌＫＩＳＡ，ＥＮＴＥＮ，ＫＩＺＹＵＮ等ＤＡＰシステムの中には含まれていないサブルーチンから構成されており，その機能は次の通りです。

ＰＲＩＮＴ（Ｎ）

描画の制御をドラフター，ラインプリンターの両方に対して行ないます。引数その他の詳細は，表２と表３を御覧下さい。このサブルーチンを活用する際には，脚注[*1]のような注意が必要です。

---

*1　ＰＲＩＮＴ(N)を引用する際には，次のような注意が必要です。

ｉ）ＰＲＩＮＴ（１）をコールする際，Ｘ，Ｙ方向の倍率は，共に１でなければならない。Ｘ，Ｙ方向の倍率が１以外の倍率（Ｘ，Ｙ）を用いたい場合には，

```
CALL FACTŌR(1.0, 1.0)
CALL PRINT(1)
CALL FACTŌR(X, Y)
```

とする必要がある。倍率が１以外で使用されると，ドラフター描画の際，隣り合う区画原点間の距離（区画の大きさ）まで変換されてしまい，区画原点が正しく設定されないためである。もしこのように誤って使った場合は，ドラフターで描画する際は，倍率の変化，原点移動をトレースできないため，エラーメッセージは出力されない。ラインプリンターを用いてシミュレートした際は，図は正しく描かれるが，エラーメッセージと変更方法が出力される。

ⅱ）ＰＲＩＮＴ（１）とともに，サブルーチンＧＥＮＴＥＮをコールして，描画原点を移動すると，ＰＲＩＮＴの内でもＧＥＮＴＥＮを使用しているため描画原点の移動が正しくは行なわれない。この場合は，ＣＡＬＬ　ＧＥＮＴＥＮ（Ｘ，Ｙ）のかわりに，ＣＡＬＬ ＥＮＴＥＮ（Ｘ，Ｙ）として使用すればよい。これが行なわれていないときは，ドラフターでは原点移動は正しく行なわれないし，エラーメッセージも出力されない。ラインプリンターでシミュレートする際は，正しく描画されるが，エラーメッセージも出力される。

表2.　PRINT(N)の引数の説明

引数N ＼ 描画用出力装置	ラインプリンター	ドラフター	
1	画面を印刷[*1]して，クリア	区画を変更（第i区画→第（i＋1）区画）[*2]	
−2	画面を印刷[*1]して，クリアし右下に「KASANE」と印刷	無視	
2[*3]	印刷せず，クリア	サブルーチンNŌSEQをコール[*4]	
4 6	無視	区画数をNに設定し第1区画を設定	x方向（横方向）の区画数を2に設定
9			x方向の区画数を3に設定
5[*3]	サブルーチンNŌSEQを使用することを設定		

＊1　画面に何も描かれていないときは，その旨出力して，画面の印刷は省略する。

＊2　あらかじめ設定された区画数を越えたときは，その旨出力し，第1区画を設定する。

＊3　現在，阪大センターのドラフターシステムでは，サブルーチンNŌSEQとGENTENの両方を引用しているとき，正しく原点の移動等が行なわれないが，近い将来改善されることを期待して当システムは作成してある。これは，引数Nをコントロールするだけで，不要な図を省略する時に用いる。

＊4　PRINT(5)によるサブルーチンNŌSEQ使用の設定が行なわれていないときにも，NŌSEQがコールされる。

表3.　DS−1やDS−2がPRINT(N)を引用している場合の引数の値とその機能

引数N	引用しているサブルーチン名	ラインプリンター	ドラフター
0	ENTEN	無視	各区画座標系において描画原点を正しく設定
8	ŌŌKISA	無視	ドラフター座標系において，隣り合う区画原点間の距離（ドラフター座標系で㎜単位）を変更
−10	DAPSTR	ラインプリンターによる描画であることの設定	
−1	DAPEND	画面を印刷[*1]する	

＊1　画面に何も描かれていないときは，その旨出力して，画面の印刷は省略する。

ŌŌKISA(XXL, YYL)

隣り合う区画原点間の距離（区画の大きさ）の変更を行ないます。XXL，YYLは，隣り合う区画の横方向（X方向），縦方向（Y方向）の区画原点間の距離です。

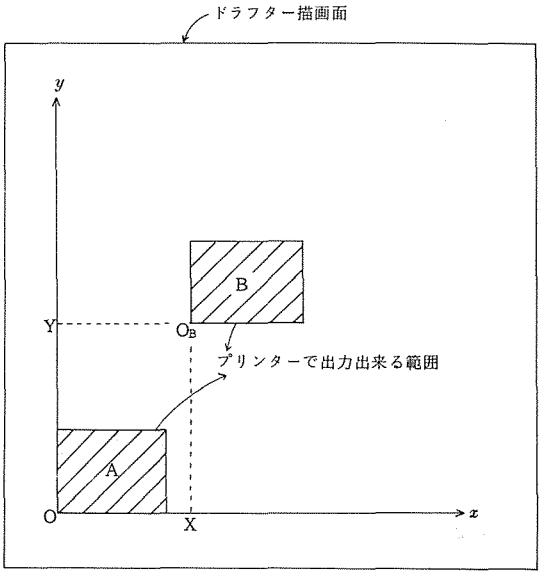
ENTEN(X, Y)

ドラフター座標系またはプリンター座標系に対して，描画原点を正しく設定します。X，Yは，仮想座標系に対する新描画座標系の原点です。PRINT(N)の脚注(ii)のような場合に用います。

KIZYUN(X, Y)

プリンター基準点を設定します。X，Yは，プリンター基準点で，仮想座標系において，プリンター原点と一致する点の座標です。

このサブルーチンにより，ドラフターで大きな図を描く場合は，ディバック時には図6のように，その任意の場所をラインプリンターで，図化させて出力することができますが，大きさは，プリンター用紙一頁分の大きさに限定されます。



xOy：ドラフター座標系　（O：初期原点）
A：初期設定時のラインプリンターでの描画範囲，初期原点と一致する
O：初期設定時のプリンター基準点
B：サブルーチンKIZYUN(X, Y) が、コールされたときの
　　ラインプリンターでの描画範囲
O_B：サブルーチンKIZYUN(X, Y) が、コールされたときの
　　プリンター基準点

図6.　ラインプリンターでの描画範囲

4. DS－2：DAPシミュレーター

ＤＳ－２に含まれるサブルーチンは，次のとおりです。

ＭＡＲＫ（Ｘ，Ｙ，ＩＢＣＤ）

記号を画面の配列に割りふるサブルーチンで，（Ｘ，Ｙ）は，記号の打たれるべき位置を，ＩＢＣＤは，記号が入っている変数名またはリテラル定数をあらわします。

ＭＡＲＫがコールされた場合，記号の中心が位置する点を含む画面の配列のます目に，サブルーチンＮＴＲを用いて，記号を割りふります。大きさ，向きは，ラインプリンターの活字で代用するため一定です。

ＮＴＲ（Ｉ，Ｊ，ＫＩＧＯ̄Ｎ）

ＮＥＡＣシリーズ２２００（７００）では，１ワード＝４８ビット＝８キャラクタで，１変数（１ワード）には，８文字の記号が格納可能です。このことをうまく使えば，文字を記憶する画面の配列ＮＧのディメンジョンを（５６，１３０）から（５６，１７）へと，約1/8にへらすことができます。（図３参照）

ＮＴＲは，この記憶容量の縮少をはかるため，ビット関数を用いてキャラクタ単位で文字を処理するサブルーチンです。（Ｉ，Ｊ）は，記号が割りふられるます目の位置，ＫＩＧＯ̄Ｎは記号をあらわします。

ここで用いたビット関数は，ＮＥＡＣシリーズ２２００（７００）の基本外部関数の１つであるＩＴＲＦＯ̄Ｍ（Ｋ，Ｌ，Ｉ，Ｊ，Ｎ）で，Ｋの値のＩ番目のビットからＮビットを，Ｌの値のＪ番目のビットからＮビットで置きかえる働きをしますが，詳しくは，文献（５）の表８－２　基本外部関数の項を御覧下さい。

K ：	k_1，k_2，……，k_{I-1}，k_I，……，k_{I+N-1}，k_{I+N}，……，k_{48}
L ：	l_1，l_2，……，l_J，……，l_{J+N-1}，……，l_{47}，l_{48}
ITRFO̅M：	k_1，……，k_{I-1}，l_J，……，l_{J+N-1}，k_{I+N}，……，k_{48}

ここで，上の□内は，内部表現であり，k_1，……，k_{48}，l_1，……，l_{48}は０または１をあらわします。

ＬＩＮＥ１（ＸＳ，ＹＳ，ＸＦ，ＹＦ）

画面の配列上に直線を描くサブルーチンで，（ＸＳ，ＹＳ）は始点，（ＸＦ，ＹＦ）は終点です。

ラインプリンターで線分を描く際のアルゴリズムは，ラインプリンターのプリント用紙の特性から，直線の傾きの絶対値が１０／６以下なら，縦列に１個の割で，１０／６より大きい場合は，横列に１個の割で，記号たとえば「＊」を，割りふるという方式を採用しています。

ＡＲＣ１（ＸＳ，ＹＳ，ＸＦ，ＹＦ，ＸＯ̄，ＹＯ̄，Ｎ）

円弧を描くサブルーチンで，（ＸＳ，ＹＳ）は始点，（ＸＦ，ＹＦ）は終点，（ＸＯ̄，ＹＯ̄）は中心，Ｎは１なら時計回り，２なら反時計回りの円弧であることをあらわしています。

円弧を描くには，中心（x_0，y_0），半径ｒとして，

$x = x_0 + r \cdot \cos\theta$

$y = y_0 + r \cdot \sin\theta$

により，θ を媒介変数として変化させ，（ x ， y ）に相当する位置に，サブルーチン MARKを用いて記号を割りふっています。

CIRC1（XŌ，YŌ，R）

円を描くサブルーチンで，（XŌ，YŌ）は中心，Rは半径です。

計算時間の短縮のため，対称性を利用すると共に，図7の㋑の範囲の円弧については，横列に1個の割で，㋺の範囲については，縦列に1個の割で，記号を割りふる方式を採用してあります。

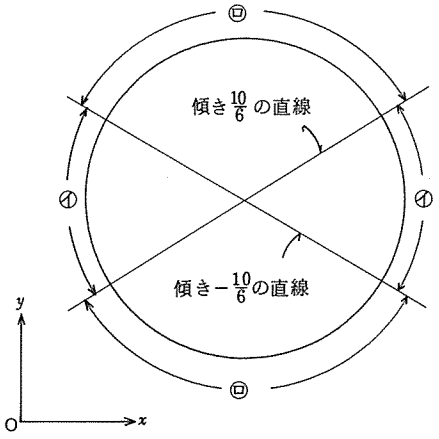


図7. 円（サブルーチンCIRC1）の描画

（㋑の範囲の円弧については横列に1個
㋺の範囲については縦列に1個の割で記号を割りふる）

DAPSTR（NŌ，N）

ラインプリンターで描画する場合の初期設定を行ないます。NŌ，Nは，DAPシステムに準ずるための引数であり，意味はありません。

このサブルーチンは，暗黙のうちにラインプリンターを描画用の出力装置として用いることを示す役割も果たしています。

LINE2（XS，YS，XF，YF，S1，S2，K）

LINE1とは異なる種類の直線を描きます。（XS，YS）は始点，（XF，YF）は終点です。記号は，直線の傾きにより，3列または3行に1個の割で，画面の配列に割りふられます。ラインプリンターによる図の解像力が悪いため，破線，一点鎖線，二点鎖線の区別はしていません。したがって，S1，S2，Kは，DAPシステムに準ずるための引数で，特別の意味はありません。

DAPEND

RŌT，ARRŌW，ARRŌWL，CURVX等の無視するサブルーチンを処理するとともに，画面の配列を図として出力します。

なお，RŌT，INCHEGについては，描画全体に影響を及ぼすので，コールされるごとに，コメントを打ち出します。その他のサブルーチンについては，DAPENDがコールされたときに，まとめてコメントと共にコールされた回数が出力されます。

これら以外に，PEN，APEN[*1]，GENTEN，FACTŌR，PLŌT等のサブルーチンが使用可能ですが，DAPシステムに準じてあるので詳細は省略します。

5. DAPシミュレーターのプログラム例と使用法

図8と図9は，DAPシミュレーターのプログラム例です。プログラム・ステートメント数は，約380（DS－1が約110，DS－2が約270）で，記憶容量は約50 kch.（約6400ワード）です。

プログラムは，他の計算機との互換性と読みやすさを考慮して，できるだけベーシック・フォートランの範囲で作成してあります。しかし，すでに述べたように，記憶容量の節約のためビット関数を，プリンター用紙の節約のためEntry文をそれぞれ用いています。

共通ブロック内の配列と変数の内容は，表4を御覧下さい。

＊1　PEN（K），APEN（K，IT）は，ペン選択番号Kに対応して，異なる記号（＊，×，＋，・，－，I）を選択します。

表 4. コモン文中の配列と変数の説明

ブロック名	変数名(配列名)＊1	説　　明	初　期　値＊2	描画用出力装置
GC	NGC	コメント制御変数 負ならコメントをほとんど出力しない 0なら主なコメントのみ出力する 正ならすべてのコメントを出力する 詳しくは各サブルーチン（図8.図9）参照	0	ラインプリンター ドラフター
GŌ	XGŌ, YGŌ	仮想座標系上での描画原点 サブルーチンENTENで変更される	0, 0	ドラフター ＊3 (ラインプリンター)
GL	XLL, YLL	1区画の大きさ(隣り合う区画原点間の距離)をサブルーチン ŌŌKISA からサブルーチンPRINTに移すために用いられる変数。	不　定	
GZ	XKZY, YKZY	プリンター基準点(仮想座標系上でプリンター原点と一致する点)の座標	0, 0	ラインプリンター
GN	NG(56,17)	56×130のます目をもつ画面の配列	すべての要素 8H△△△△△△△△	
GK	KIGŌ	ラインプリンターで線をあらわすために用いる記号，サブルーチンPEN, APENで変更される	1H＊	
GM	IMAX, JMAX	ます目の数をあらわす変数	56, 130	
GT	NTC	画面に記号を代入した回数	0	
GR	XRATE, YRATE	倍　　率	1, 1	
GG	XG, YG	プリンタ座標系における描画原点	0, 0	
GX	XJ, YI	ます目1字分の大きさ	2.54, 2.54/0.6	
G2	IJP	ラインプリンターで直線をあらわすとき何行目か何列目ごとに記号をプロットするかを示す変数 サブルーチンLINE2からLINE1をコールする際3に変更され，リターンして，LINE2にもどってきたとき，1にもどされる	1	
GP	XP, YP	サブルーチンPLŌT用のプリンター座標系における現在点の座標をあらわす変数 プリンタ座標系上の値として設定される	0, 0	

＊1 各サブルーチンで，同じ変数名，配列名を用いている。

＊2 ラインプリンターで描画する際サブルーチンDAPSTRをコールしたとき設定される値を示す。ドラフターで描画する際は，初期値設定は行なわないが，NEACシリーズ2200(700)ではコモン文中の 変数に対しては必ず，初期値は0となるのでこれを用いているのもある(NGC, XGŌ, YGŌ)。

＊3 エラーのチェックに用いられる。

```
ISN    LABEL FORTRAN STATEMENT

       C*****   DAP SIMULATOR
       C        DS-1 (DRAFTA,LINE PRINTER)        〔描画の制御を行なうサブルーチンセット DS−1〕
       C*****
0001          SUBROUTINE PRINT(N)                ┐ 0. 描画を制御するサブルーチン。仮引数Nは，描画制御変数。
0002          COMMON /GN/NG(56,17)               │
            $    /GO/XGO,YGO /GL/XLL,YLL         │ 1. コモン文の内容は，表4参照。
            $    /GC/NGC /GM/IMAX,JMAX           │
            $    /GT/NTC /GR/XRATE,YRATE         │
            $    /GG/XG,YG /GZ/XKZY,YKZY         ┘
0003          DATA L,LP/1,0/,KP,KP2/2,4/         ┐ 2. データ文中の変数の説明は，表5参照。
            $    ,XL,YL/350.0,250.0/             │
            $    ,KUHAKU/8H        /             │
            $    ,NOS,NOSW/0,0/                  ┘
0004          IF(N.NE.-10) GO TO 35              ┐ 3. 描画制御変数Nが−10のとき，ラインプリンターによる描画であること
0005          L=-10                              │ とを設定し，リターン。
0006          RETURN                             ┘
0007       35 IF(N.NE.5) GO TO 40                ┐ 4. Nが5のとき，サブルーチンNŌSEQを使用することを設定し，リ
0008          NOSW=1                             │ ターン。
0009          RETURN                             ┘
0010       40 IF(L.EQ.-10) GO TO 120             ┐ 5. 描画用出力装置の判定を行なう。（ラインプリンターなら18.へとぶ。）

       C     *** DRAFTA  OUTPUT ***               〔ドラフターによる描画用ルーチン〕
0011          IF(NGC.GE.0)  WRITE(6,55) N        ┐ 6. コメント制御変数が0か正なら，コメントを出力する。
0012       55 FORMAT(1H ,5X,4H*** ,5X,           │
            $    6HPRINT(,I3,1H))                ┘
0013          IF(N.NE.8) GO TO 50                ┐ 7. Nが8なら，1区画の大きさ（隣り合う区画原点間の距離）を設定し
0014          XL=XLL                             │ なおす。
0015          YL=YLL                             │
0016          RETURN                             ┘
0017       50 IF(N.EQ.9) GO TO 142               ┐ 8. Nが9なら15.へとび，4か6なら14.へとぶ。
0018          IF(N.EQ.4.OR.N.EQ.6)               │ Nが−2ならリターン。
            $    GO TO 140                       │ Nが0なら11.へとぶ。
0019          IF(N.EQ.-2) RETURN                 │
0020          IF(N.EQ.0) GO TO 80                ┘
0021          IF(N.NE.2.AND.NOSW.EQ.0)           ┐ 9. Nが2なら，サブルーチンNŌSEQをコールしてリターン。
            $    GO TO 70                        │ また，NŌSEQの使用が設定されているとき，シーケンス・ナンバ
0022          NOS=NOS+1                          │ −NŌSを1増し，NŌSEQをコールする。
0023           WRITE(6,60) NOS                   │
0024       60 FORMAT(1H ,5X,8H**NOSEQ(,I2,       │
            $    1H))                            │
0025          CALL NOSEQ(NOS)                    │
0026          IF(N.EQ.2) RETURN                  ┘
0027       70 LP=LP+1                            ┐ 10. ドラフターで，どの区画に描くか算定する。
0028          IF(LP.LE.0.OR.LP.GE.KP2)           │ あらかじめ設定された区画数を超えたときは，12.へとぶ。
            $    GO TO 150                       │
0029          LX=LP-LP/KP*KP                     │
0030          LY=LP/KP                           ┘
0031       80 XO=FLOAT(LX)*XL+XGO                ┐ 11. 新しく設定された区画座標系に対して描画原点を設定しなおし，コメ
0032          YO=FLOAT(LY)*YL+YGO                │ ントを出力し，リターン。
0033           WRITE(6,85) XO,YO                 │
0034       85 FORMAT(1H ,5X,11H*** GENTEN(,      │
            $    F12.6,1H,,F12.6,1H))            │
0035          CALL GENTEN(XO,YO)                 │
0036          RETURN                             ┘
0037      150  WRITE(6,160) LP                   ┐ 12. あらかじめ設定された区画数を越えたことと，描画用紙の交換を要求
0038      160 FORMAT(1H0,5X,12H***OVER***        │ するメッセージを出力する。
            $    ,14H  PRINT DE LP=,I6,          │
            $    19HDESU NODE ATARASII ,         │
            $    21HKAMI NI KAETE KUDASAI)       ┘
0039      180 CALL HALT                          ┐ 13. サブルーチンHALTをコールするとともに，新しい画面における第
0040          LP=0                               │ 1区画を設定し，描画原点を新しい区画座標系に対して設定しなおす。
0041           WRITE(6,85) XGO,YGO               │
0042          CALL GENTEN(XGO,YGO)               │
0043          RETURN                             ┘
0044      140 KP=2                               ┐ 14. 区画数をN（4か6）に設定する。また，x方面（横方向）の区画数
0045          KP2=N                              │ を2に設定する。
0046          GO TO 143                          ┘
0047      142 KP=3                               ┐ 15. 区画数を9に設定する。また，x方向の区画数を3に設定する。
0048          KP2=N                              ┘
0049      143 IF(LP.NE.0) GO TO 144              ┐ 16. 新しく設定しなおした区画数を出力し，13.へとぶ。
0050           WRITE(6,147) N                    │
0051      147 FORMAT(1H ,5X,10H ***NEW***,       │
            $    33X,14HGAMEN NO KAZU=,          │
            $    I2)                             │
0052          GO TO 180                          ┘
0053      144  WRITE(6,146) N                    ┐ 17. 新しい画面を設定したこと，およびその区画数と，描画用紙の交換を
0054      146 FORMAT(1H0,5X,10H***NEW*** ,       │ 要求するメッセージを出力する。
            $    20H   ATARASII KAMI NI ,        │
            $    17HKAETE KUDASAI     ,          │
            $    14HGAMEN NO KAZU=,I12)          │
0055          GO TO 180                          ┘
```

図8.（その1） DS−1のプログラム

```
C     *** LINE PRINTER OUTPUT ***
0056  120 IF(N.EQ.4.OR.N.EQ.9.OR.
        $    N.EQ.6.OR.N.EQ.0.OR.N.EQ.8
        $    ) RETURN
0057        IF(N.EQ.2) GO TO 133
0058        IF(NTC.NE.0) GO TO 123
0059        IF(N.EQ.-1) RETURN
0060         WRITE(6,122)
0061  122 FORMAT(1H0,100X,8H=HAKUSI ,
        $    5HDESU=)
0062        GO TO 128
0063  123  WRITE(6,125) ((NG(I,J),
        $    J=1,17),I=1,IMAX)
0064  125 FORMAT(1H1,16A8,A2/(1H ,16A8,
        $    A2))
0065        IF(NGC.GE.0)  WRITE(6,126)
0066  126 FORMAT(1H+,1H+)
0067        IF(NGC.GE.0)  WRITE(6,127)
        $    NTC
0068  127 FORMAT(1H ,100X,4HNTC=,I6)
0069        NTC=0
0070        IF(N.EQ.-1) RETURN
0071  128 IF(N.EQ.-2)  WRITE(6,130)
0072  130 FORMAT(1H ,126X,6HKASANE)
0073        IF(N.EQ.-2) GO TO 135
0074        IF(XRATE.NE.1.0.OR.YRATE.NE.
        $    1.0)  WRITE(6,129) N,
        $    XRATE,YRATE
0075  129 FORMAT(1H0,12H***OMEGAI***,
        $    6X,19HBAIRITU GA (1.0,1.0
        $    ,16H) DE NAITOKI NI ,
        $    20H<PRINT> O CALL SITE ,
        $    19HIMASU NODE TUGI NO ,
        $    18HYCONT HENKOO SITE ,
        $    7HKUDASAI/1H ,20X,4HCALL ,
        $    15HFACTOR(1.0,1.0)/1H ,
        $    20X,11HCALL PRINT(,I3,1H)
        $    /1H ,20X,12HCALL FACTOR(,
        $    F12.6,1H,,F12.6,1H)//)
0076        IF(XGO-XKZY.NE.XG.OR.YGO-YKZY
        $    .NE.YG)  WRITE(6,132)
0077  132 FORMAT(1H0,12H***OMEGAI***,6X
        $    ,15H<CALL GENTEN> O,
        $    23H <CALL  ENTEN> NI KAETE
        $    ,8H KUDASAI/1H ,18X,4HMATA
        $    ,22H BAIRITU GA (1.0,1.0)
        $    ,22HDE NAI TOKI NIWA TYUUI
        $    ,16H GA HITUYOO DESU)
0078        IF(NOSW.EQ.0) GO TO 135
0079  133 NOS=NOS+1
0080         WRITE(6,60) NOS
0081        CALL NOSEQ(NOS)
0082  135 DO 400 I=1,IMAX
0083        DO 400 J=1,17
0084  400 NG(I,J)=KUHAKU
0085        RETURN

0086        ENTRY KIZYUN(X,Y)
C             COMMON /GS//G7/
0087        XKZY=X
0088        YKZY=Y
0089        XG=XG-XKZY
0090        YG=YG-YKZY
0091        IF(L.EQ.-10)  WRITE(6,500)
        $    XKZY,YKZY
0092  500 FORMAT(1H ,6X,10H==KIZYUN==,
        $    5X,1H(,F12.6,1H,,F12.6,1H)
        $    )
0093        RETURN
0094        END

C*****
0001        SUBROUTINE OOKISA(XXL,YYL)
0002        COMMON /GL/XLL,YLL
0003        XLL=XXL
0004        YLL=YYL
0005        CALL PRINT(8)
0006        RETURN
0007        END
```

〔ラインプリンターによる描画用ルーチン〕

18. Nが4, 9, 6, 0, 8のいずれかのとき，リターン。
Nが2のとき，26.へとぶ。

19. Nが－1のとき，リターン。
画面に何も描かれなかったとき，その旨を出力し，23.へとぶ。

20. 画面の出力を行なう。

21. コメント制御変数が0か，正のとき，プリンタ原点に「＋」をプロットし，また画面に代入された記号の数を出力するとともに，その数をクリアする。

22. Nが－1ならリターン。

23. Nが－2のとき，画面の右下に「重ね」である旨を出力し，27.へとぶ。

24. X，Y方向の倍率が1でないとき，このサブルーチンＰＲＩＮＴがコールされていることと，その変更の方法を示すメッセージを出力する。

25. 利用者のプログラムで，ＰＲＩＮＴ，ＧＥＮＴＥＮの両方のサブルーチンをコールしているとき，注意を換起する旨，出力する。

26. サブルーチンＮＯＳＥＱの使用が設定されているとき，シーケンス・ナンバーを1増し，ＮＯＳＥＱをコールする。

27. 画面をクリアし，リターン。

28. 以下，プリンター基準点を設定するサブルーチン。

29. プリンター基準点を変更し，プリンター原点を設定しなおす。

30. ラインプリンターによる描画のとき，コメントを出力する。

31. リターン。

0. 1区画の大きさ（隣り合う区画原点間の距離）を変更するサブルーチン。

1. ＰＲＩＮＴ（8）をコールし，隣り合う区画原点間の距離を設定しなおし，リターン。

図8（その2）　ＤＳ－1のプログラム

```
C*****
      SUBROUTINE  ENTEN(XE,YE)
      COMMON /GO/XGO,YGO
      XGO=XE
      YGO=YE
      CALL GENTEN(XE,YE)
      CALL PRINT(0)
      RETURN
      END
```

0. ENTENというサブルーチン。

1. PRINT(0)をコールし，各座標系（ドラフターなら，区画座標系，ラインプリンターならプリンター座標系）に対して描画原点を正しく設定しなおす。そして，リターン。

図8（その3）　DS－1のプログラム

表5. サブルーチンPRINTの中のDATA文の説明

変数名	説明	初期値
L	描画用出力装置判定変数　（1ならドラフターを，－10ならラインプリンターをあらわす　）	1
LP	区画を示す変数（第（LP＋1）区画をあらわす）	0
KP	横方向（X方向）の区画数をあらわす変数	2
KP2	画面内の全区画数をあらわす変数	4
XL，YL	区画の大きさ（隣り合う区画原点間の距離）をあらわす変数	350.0，250.0
KUHAKU	画面の配列をクリアするための文字定数	8H△△△△△△△△
NŌS	サブルーチンNŌSEQ用のシーケンス・ナンバー	0
NŌSW	サブルーチンNŌSEQを使用するか否かを示す変数（0なら使用しないことをあらわし，1は使用することをあらわす）	0

```
ISN    LABEL FORTRAN STATEMENT

       C*****   DAP SIMULATOR                          〔ラインプリンターを用いて描画するときだけ入れるサブルーチンセットDS－2〕
       C        DS-2 (LINE PRINTER)
       C*****
0001          SUBROUTINE DAPSTR(NO,N)                  0. DAPSTRというサブルーチン。
       C   ENTRY HALT,PEN(K),APEN(K,IT),                 （ラインプリンターによる描画用に初期設定を行なう。）
       C      GENTEN(XO,YO),MARK(X,Y,IBCD),
       C      NOSEQ(N),FACTOR(X,Y)
0002          COMMON /GN/NG(56,17)                     1. コモン文の内容については，表4参照
            $    /GM/IMAX,JMAX /GG/XG,YG                  （以下のサブルーチン中のコモン文についても同様）
            $    /GK/KIGO /GX/XJ,YI /GC/NGC
            $    /GR/XRATE,YRATE /GZ/IJP
            $    /GZ/XKZY,YKZY /GT/NTC
0003          DIMENSION KI(6)                          2. 初期値として，ペン選択番号に対応する記号を設定する。また，KUHAKUはブランクを入れる文字定数。
0004          DATA KI/1H*,1HX,1H+,1H.,1H-,
            $    1HI/,KUHAKU/8H        /

       C        COMMON /GN//GK//GM//GX//GR/              （この種類の注釈行については8章参照のこと。）
       C           /G2//GC//GG//GO/
       C        DATA KI,KUHAKU
0005          DO 40 I=1,56                             3. 画面の配列をクリアする。
0006          DO 40 J=1,17
0007       40 NG(I,J)=KUHAKU
0008          KIGO=KI(1)                               4. 初期値を設定する。
0009          IMAX=56                                     詳しくは，表4参照。
0010          JMAX=130                                    PRINT(－10)をコールして，ラインプリンターを描画用の出力装置として用いることを設定する。
0011          XJ=2.54
0012          YI=XJ/0.6
0013          XRATE=1.0
0014          YRATE=1.0
0015          NGC=0
0016          XG=0.0
0017          YG=0.0
0018          IJP=1
0019          NTC=0
0020          XKZY=0.0
0021          YKZY=0.0
0022          XGO=0.0
0023          YGO=0.0
0024          CALL PRINT(-10)
0025           WRITE(6,45) NO,N                        5. コメントを出力し，リターン。
0026       45 FORMAT(1H0,6X,10H==DAPSTR==,5X
            $    ,1H(,I2,1H,,I3,1H))
0027          RETURN

0028          ENTRY HALT                               6. 以下，HALTというサブルーチン。
       C        COMMON /GC/
0029          IF(NGC.GE.0)  WRITE(6,47)                7. コメント制御変数が0か正のとき，コメントを出力する。
0030       47 FORMAT(1H ,5X,8H*** HALT)                8. リターン。
0031          RETURN
0032          ENTRY PEN(K)                             9. 以下，PENというサブルーチン。
0033          IT=1
0034          ENTRY APEN(K,IT)                         10. 以下，APENというサブルーチン。
       C        COMMON /GK//GC/
       C        DATA KI
0035          IF(K.GE.1.AND.K.LE.6)                    11. 仮引数Kが1～6以外のとき，エラーメッセージを出力し，ペン選択番号に1を設定して続行する。
            $    GO TO 60
0036           WRITE(6,50)
0037       50 FORMAT(1H0,10X,10HPEN SELECT,
            $    9H ERROR***)
0038          K=1
0039       60 KIGO=KI(K)                               12. ペン選択番号に相当する記号を設定する。
0040          IF(NGC.LT.0) RETURN                      13. コメント制御変数が負のとき，リターン。
0041           WRITE(6,70) K,KIGO                      14. コメントを出力する。
0042       70 FORMAT(1H ,6X,7H==PEN==,5X,I3
            $    ,5X,1H(,A1,1H))
0043          IF(IT.GE.0)  WRITE(6,75)                 15. 仮引数ITが0か正なら，三角形を描く旨出力する。
0044       75 FORMAT(1H+,40X,11HSANKAKKEI O
            $    ,10H EGAKIMASU)
0045          RETURN                                   16. リターン。

0046          ENTRY GENTEN(XO,YO)                      17. 以下，原点の移動を行なうサブルーチン。
       C        COMMON /GC//GG//GR//GZ/
0047          IF(NGC.GE.0)  WRITE(6,80) XO,            18. コメント制御変数が0か正のとき，コメントを出力する。
            $    YO
0048       80 FORMAT(1H ,6X,10H==GENTEN==,
            $    5X,1H(,F12.6,1H,,F12.6,1H)
            $    )
0049          XG=XO*XRATE-XKZY                         19. プリンター座標系に対して，描画原点を設定して，リターン。
0050          YG=YO*YRATE-YKZY
0051          RETURN
```

図9（その1） DS－2のプログラム

```
0052          ENTRY MARK(X,Y,IBCD)
C               COMMON /GM/GC/GR/
0053          I=FLOAT(IMAX)-(YG+Y*YRATE)
            $   /YI+0.5
0054          J=(XG+X*XRATE)/XJ+1.5
0055          IF(I.GE.1.AND.I.LE.IMAX.AND.
            $   J.GE.1.AND.J.LE.JMAX)
            $   CALL NTR(I,J,IBCD)
0056          RETURN

0057          ENTRY NOSEQ(N)
C               COMMON /GC/
0058          IF(N.LE.0.OR.N.GT.99)
            $   GO TO 100
0059          IF(NGC.GE.0)  WRITE(6,90) N
0060       90 FORMAT(1H ,5X,10H*** NOSEQ(,
            $   I2,1H))
0061          RETURN
0062      100 WRITE(6,110) N
0063      110 FORMAT(1H0,14H ***MATIGAI***,
            $   15H NOSEQ(N) DE N=,I6,
            $   4HDESU)
0064          RETURN

0065          ENTRY FACTOR(X,Y)
C               COMMON /GP//GC/
0066          IF(X.NE.0.0) GO TO 130
0067           WRITE(6,125)
0068      125 FORMAT(1H0,5X,11HXPATE EPROR)
0069          X=1.0
0070      130 IF(Y.NE.0.0) GO TO 140
0071           WRITE(6,135)
0072      135 FORMAT(1H0,5X,11HYRATE EPROR)
0073          Y=1.0
0074      140 XRATE=X
0075          YRATE=Y
0076          IF(NGC.GE.0)  WRITE(6,145)
            $   XRATE,YRATE
0077      145 FORMAT(1H ,6X,10H==FACTOR==,
            $   5X,1H(,F12.6,1H,,F12.6,1H)
            $   )
0078          RETURN
0079          END
```

20. 以下，MARKというサブルーチン。
21. 座標X，Yに対して，それぞれに対応する画面上の位置J，I を計算する。
22. 画面内なら，画面のその位置に，仮引数IBCDを入れる。
23. リターン。
24. 以下，NOSEQというサブルーチン。
25. Nが2桁以内の自然数でないとき，28へとぶ。
26. コメント制御変数が0か正のとき，コメントを出力する。
27. リターン。
28. エラーメッセージを出力して，リターン。
29. 以下，拡大・縮少を行なうサブルーチン。
30. X方向の倍率に0が指定されたとき，エラーメッセージを出力し，1として続行する。
31. Y方向の倍率に0が指定されたとき，エラーメッセージを出力し，1として続行する。
32. X，Y方向の倍率を設定しなおす。
33. コメント制御変数が0か正なら，コメントを出力する。
34. リターン。

```
C*****
0001          SUBROUTINE NTR(I,J,KIGOU)
0002          COMMON /GM/NG(56,17)
            $   /GT/NTC
0003          J2=(J+7)/8
0004          KK=(J-J2*8+7)*6+1
0005          NG(I,J2)=ITRFOMTNG(I,J2),
            $   KIGOU,KK,1,6)
0006          NTC=NTC+1
0007          RETURN
0008          END
```

0. 画面の配列内に，キャラクタ単位で文字を入れるサブルーチン
1. 画面の配列の1行内で何ワード目かを計算する。
2. 1ワード内で何キャラクタ目かを計算する。
3. 画面の配列の対応するキャラクタの部分に文字を入れて，画面の配列内への代入回数をカウントし，リターン。

```
C*****
0001          SUBROUTINE LINE2(XS,YS,XF,YF,
            $   S1,S2,K)
0002          COMMON /GC/NGC /G2/IJP
0003          IF(NGC.GE.0)  WRITE(6,690) S1
            $   ,S2,K
0004      690 FORMAT(1H ,6X,9H==LINE2==,60X
            $   ,3HS1=,F12.6,5X,3HS2=,
            $   F12.6,5X,2HK=,I3)
0005          IF(K.GE.1.AND.K.LE.3)
            $   GO TO 750
0006           WRITE(6,700) K
0007      700 FORMAT(1H0,14H***MATIGAI***,
            $   22H LINE2(XS,YS,XF,YF,S1,,
            $   11HS2,K) DE K=,I6,5H DESU)
0008          RETURN
0009      750 IJP=3
0010          CALL LINE1(XS,YS,XF,YF)
0011          IJP=1
0012          RETURN
0013          END
```

0. LINE2というサブルーチン。
1. コメント制御変数が0か正なら，コメントを出力する。
2. 仮引数Kが1～3でなければ，エラーメッセージを出力し，リターン。
3. 3列または3行に1個の割で，記号を割りふって，直線を描き，リターン。

図9（その2） DS－2のプログラム

```
C*****
      SUBROUTINE LINE1(XS,YS,XF,YF)
      COMMON /GG/XG,YG /GX/XJ,YI
     $   /GK/KIGO /GM/IMAX,JMAX
     $   /G2/IJP /GP/XP,YP /GC/NGC
     $   /GR/XRATE,YRATE
      MJX(X)=(XG+X*XRATE)/XJ+1.5
      MIY(Y)=FLOAT(IMAX)-(YG+YRATE
     $   *Y)/YI+0.5
      XP=XF*XRATE+XG
      YP=YF*YPATE+YG
      IF(NGC.LE.0) GO TO 710
       WRITE(6,700) XS,YS,XF,YF
  700 FORMAT(16X,1H(,F12.6,1H,
     $   F12.6,3H),(,F12.6,1H,,
     $   F12.6,1H))
  710 IF(XF.EQ.XS) GO TO 810
      YX=(YF-YS)/(XF-XS)
      IF(ABS(YX*YRATE/XRATE).GT.
     $   (1.0/0.6)) GO TO 810
      J1=MJX(XS)
      J2=MJX(XF)
      IF(J1.LE.J2) GO TO 720
      J3=J1
      J1=J2
      J2=J3
  720 IF(J1.GT.JMAX.OR.J2.LT.1)
     $   RETURN
      IF(J1.LT.1) J1=1
      IF(J2.GT.JMAX) J2=JMAX
      Y=YS+((FLOAT(J1-1)*XJ-XG)
     $   /XRATE-XS)*YX
      DY=FLOAT(IJP)*XJ/XRATE*YX
      DO 730 J=J1,J2,IJP
      IY=MIY(Y)
      IF(IY.GE.1.AND.IY.LE.IMAX)
     $   CALL NTP(IY,J,KIGO)
      Y=Y+DY
  730 CONTINUE
      RETURN
  810 XY=(XF-XS)/(YF-YS)
      I1=MIY(YS)
      I2=MIY(YF)
      IF(I1.LE.I2) GO TO 820
      I3=I1
      I1=I2
      I2=I3
  820 IF(I1.GT.IMAX.OR.I2.LT.1)
     $   RETURN
      IF(I1.LT.1) I1=1
      IF(I2.GT.IMAX) I2=IMAX
      X=XS+((FLOAT(IMAX-I1)*YI-YG)
     $   /YRATE-YS)*XY
      DX=-FLOAT(IJP)*YI/YRATE*XY
      DO 830 I=I1,I2,IJP
      JX=MJX(X)
      IF(JX.GE.1.AND.JX.LE.JMAX)
     $   CALL NTP(I,JX,KIGO)
      X=X+DX
  830 CONTINUE
      RETURN
      END
```

0. 直線を描くサブルーチン。

1. 座標X，Yに対して，それぞれに対応する画面上の位置J，Iを算出する文関数。

2. 現在点（サブルーチンPLOT用）として，直線の終点をプリンタ座標系上の値として設定する。

3. コメント制御変数が，正のとき，コメントを出力する。

4. 描く直線の傾きの絶対値が10/6より大きいとき，10.へとぶ。

5. 始点，終点のX座標に対する画面上の位置を1.の文関数を用いて算出する。

6. 始点のX座標が，終点より右にあるとき，J1とJ2を入れかえる。

7. 直線が画面の外にあるとき，リターン。

8. 始点または終点が画面の外にあるとき，画面内だけを描くように，設定する。

9. ラインプリンターで直線を描くのに必要なすべての点の画面上での値を算出し，その点が画面内にあれば，記号を入れる。そして，リターン。

10. 以下，直線の傾きの絶対値が10/6より大きい場合について，5.～9.と同様の処理を行なう。そして，リターン。

```
C*****
      SUBROUTINE PLOT(X,Y,IP)
      COMMON /GP/XP,YP /GG/XG,YG
     $   /GR/XRATE,YRATE
       IF(IP.EQ.3) GO TO 20
       IF(IP.EQ.2) GO TO 30
        WRITE(6,10) IP
   10 FORMAT(1H ,14H***MATIGAI*** ,
     $   19HPLOT(X,Y,IP) DE IP=,
     $   I6,4HDESU)
      RETURN
   20 XP=X*XRATE+XG
      YP=Y*YRATE+YG
      RETURN
   30 XPP=(XP-XG)/XRATE
      YPP=(YP-YG)/YRATE
      CALL LINE1(XPP,YPP,X,Y)
      RETURN
      END
```

0. PLOTというサブルーチン。

1. 仮引数IPが3なら3.へ，2なら4.へとぶ。

2. エラーメッセージを出力し，リターン。

3. 現在点をプリンタ座標系上の値として設定しなおし，リターン。

4. サブルーチンLINE1をコールし，現在点と点（X，Y）を直線で結ぶ。そして，リターン。

図9（その3） DS－2のプログラム

```
      C*****
0001        SUBROUTINE ARC1(XS,YS,XF,YF,
           $XO,YO,N)
0002        COMMON /GK/KIGO /GX/XJ,YI
           $   /GC/NGC /GR/XRATE,YRATE
      C     N=1:TOKEI HOOKOO
      C     N=2:HANTOKEI HOOKOO
0003        IF(N.EQ.1.OR.N.EQ.2)
           $   GO TO 110
0004          WRITE(6,120)
0005    120 FORMAT(1H0,5X,10HARC1 ERROR)
0006        N=1
0007    110 IF(NGC.GT.0)  WRITE(6,140)
           $   XS,YS,XF,YF,XO,YO,N
0008    140 FORMAT(1H ,5X,9H*** ARC1(,
           $   6(F12.6,1H,),I6,1H))
0009        IF(XRATE.EQ.YRATE) GO TO 130
0010          WRITE(6,120)
0011        CALL FACTOR(1.0,1.0)
0012    130 RS=SQRT((XS-XO)**2+(YS-YO)**2
           $   )
0013        RF=SQRT((XF-XO)**2+(YF-YO)**2
           $   )
0014        RSA=ABS(RF-RS)
0015        IF(RSA.LE.1.0) GO TO 180
0016          WRITE(6,160) RSA
0017    160 FORMAT(1H0,5X,9HARC DATA ,
           $   5HERROR,5X,9HHANKEI NO,
           $   4H SA=,F12.6,5H DESU)
0018        RETURN
0019    180 RC=(RS+RF)/2.0
0020        WS=ATAN2(YS-YO,XS-XO)
0021        WF=ATAN2(YF-YO,XF-XO)
0022        AT=WS
0023        WT=WF-WS
0024        IF(N.EQ.1) WT=-WT
0025        IF(WT.LE.0.0) WT=WT+2.0
           $   *3.14159265
0026        NI=RC/XJ*WT*ABS(XRATE)
0027        DAT=WT/FLOAT(NI)
0028        IF(N.EQ.1) DAT=-DAT
0029        CALL PLOT(XF,YF,3)
0030        KIGON=KIGO
0031        DO 190 L=1,NI
0032        X=XO+RC*COS(AT)
0033        Y=YO+RC*SIN(AT)
0034        CALL MARK(X,Y,KIGON)
0035        AT=AT+DAT
0036    190 CONTINUE
0037        RETURN
0038        END
```

0. 円弧を描くサブルーチン。
1. 仮引数Nが1,2以外なら，エラーメッセージを出力し，N＝1として続行する。
2. コメント制御変数が正なら，コメントを出力する。
3. X，Y方向の倍率が等しくなければ、エラーメッセージを出力し，それぞれの倍率が1として続行する。
4. 始点，終点から中心までの半径を計算する。
5. 半径の差が1以上のとき，エラーメッセージを出力して，リターン。
6. 半径を決定し，始点，終点の偏角を計算する。
7. 偏角の差を計算する。
8. 円弧の分割数，偏角の増分を計算する。
9. サブルーチンＰＬＯＴ用に現在点を設定する。
10. ラインプリンターで円弧を描くのに必要な点の位置を計算し，画面に記号を入れてリターン。

```
      C*****
0001        SUBROUTINE CIRC1(XO,YO,R)
0002        COMMON /GK/KIGO /GX/XJ,YI
           $   /GR/XRATE,YRATE /GC/NGC
0003        IF(NGC.GT.0)  WRITE(6,50)
           $   XO,YO,R
0004     50 FORMAT(1H ,6X,9H***CIRC1(,
           $   2(F12.6,1H,),F12.6,1H))
0005        KIGON=KIGO
      C       0.6**2+1.0**2=1.36
0006        SQ136=1./SQRT(1.36)
0007        IF(XRATE.EQ.YRATE) GO TO 70
0008          WRITE(6,80)
0009     80 FORMAT(1H0,5X,10HARC1 ERROR)
0010        CALL FACTOR(1.0,1.0)
0011     70 CALL MARK(XO,YO+R,KIGON)
0012        CALL MARK(XO,YO-R,KIGON)
0013        CALL PLOT(XO-R,YO,3)
0014        NX=R*SQ136/XJ*ABS(XRATE)
0015        IF(NX.LT.1) GO TO 150
0016        X=0.0
0017        DO 100 N=1,NX
0018        X=X+XJ
0019        DY=SQRT(R*R-X*X)
0020        CALL MARK(XO+X,YO+DY,KIGON)
0021        CALL MARK(XO+X,YO-DY,KIGON)
0022        CALL MARK(XO-X,YO+DY,KIGON)
0023    100 CALL MARK(XO-X,YO-DY,KIGON)
0024    150 CALL MARK(XO+R,YO,KIGON)
0025        CALL MARK(XO-R,YO,KIGON)
```

0. 円を描くサブルーチン。
1. コメント制御変数が正なら，コメントを出力する。
2. 記号を設定する。そして6.と9.の準備計算を行なう。
3. X，Y方向の倍率が等しくないとき，エラーメッセージを出力し，それぞれの倍率を1にして続行する。
4. 図10のイ，エの点をプロットする。
5. サブルーチンＰＬＯＴ用に現在点を設定する。
6. 図10のxnの区間の分割数を計算し，1より小さいとき，8へとぶ。
7. 図10のい，き，う，かの区間の円弧をラインプリンターで描くときの点の位置を計算し，画面に記号を入れる。
8. 図10のア，ウの点をプロットする。

図9（その4）　ＤＳ－２のプログラム

```
0026          NY=R*0.6*SQ136/YI*ABS(YRATE)
0027          IF(NY.LT.1) RETURN
0028          Y=0.0
0029          DO 200 N=1,NY
0030          Y=Y+YI
0031          DX=SQRT(R*R-Y*Y)
0032          CALL MARK(XO+DX,YO+Y,KIGON)
0033          CALL MARK(XO-DX,YO+Y,KIGON)
0034          CALL MARK(XO+DX,YO-Y,KIGON)
0035      200 CALL MARK(XO-DX,YO-Y,KIGON)
0036          RETURN
0037          END
```

9. 図10のynの分割数を計算し，1より小さいとき，リターン。

10. 図10のあ，え，く，おの区間の円弧をラインプリンターで描くのに必要な点の位置を計算し，画面に記号を入れる。そして，リターン。

```
      C*****
0001          SUBROUTINE DAPEND
0002          COMMON /GC/NGC
0003          DATA ISWT/0/
0004          IF(NGC.GE.0)  WRITE(6,200)
0005      200 FORMAT(1H ,5X,10H*** DAPEND)
0006          CALL PRINT(-1)
0007          IF(ISWT.NE.0)  WRITE(6,250)
            $    ISWT
0008      250 FORMAT(1H0,5X,10H***MUSI***,
            $    5X,19H CALL ARROW,ARROWL,,
            $    18HCURVX,DIMAN,DIMEN,,
            $    21HELIPS,FAN1,FAN2,GRID,,
            $    23HNUMBR,PARAB,POLAR,POLY,
            $    ,22HRECT,SHADE,SLINE,SMOTH
            $    1H,/21X,14HSYMBL1,SYMBL2,,
            $    20HTRIGL  WA MUSI SITE ,
            $    5HIMASU,10X,10HMUSI SITA ,
            $    5HKAZU=,I6,11H(NOBE) DESU)
0009          RETURN
```

0. DAPENDというサブルーチン。

1. ISWTは5.のサブルーチンのコール回数。

2. コメント制御変数が0か正のとき，コメントを出力する。

3. 画面に描画されているとき，出力する。

4. 5.に示すサブルーチンのいずれかが1回以上コールされたとき，無視しているサブルーチン名と共に，コールされた回数を出力する。そして，リターン。

```
0010          ENTRY ARROW(XS,YS,XF,YF,S,
            $    ICODE)
0011          ENTRY ARROWL(X,Y,N,S,ICODE)
0012          ENTRY CURVX(XS,XF,A1,A2,A3,A4
            $    ,K1,K2,K3,K4)
0013          ENTRY DIMAN(XO,YO,R,DATA,S)
0014          ENTRY DIMEN(XS,YS,DATA,S)
0015          ENTRY ELIPS(XO,YO,A,B,T1,T2,
            $    T3)
0016          ENTRY FAN1(XO,YO,R1,R2,A1,A2)
0017          ENTRY FAN2(X1,Y1,X2,Y2,X3,Y3,
            $    X4,Y4)
0018          ENTRY GRID(XS,YS,X,Y,M,N,S)
0019          ENTRY NUMBR(XS,YS,X,Y,M,N,S)
0020          ENTRY PARAB(XA,YA,XB,YB,XC,YC
            $    )
0021          ENTRY POLAR(R,A,N,K,IBCD)
0022          ENTRY POLY(XS,YS,WK,N,S)
0023          ENTRY RECT(XS,YS,H,W,S)
0024          ENTRY SHADE(X1,Y1,X2,Y2,D,S,
            $    N1,K1,N2,K2)
0025          ENTRY SLINE(X,Y,N,K,ITIPE,
            $    ICODE)
0026          ENTRY SMOTH(X,Y,N)
0027          ENTRY SYMBL1(XS,YS,H,ICODE,S)
0028          ENTRY SYMBL2(XS,YS,H,IBCD,S,N
            $    )
0029          ENTRY TRIGL(XS,YS,W,H,T,S)
0030          ISWT=ISWT+1
0031          RETURN
```

5. これらのサブルーチンがコールされたときは，無視される。ただし，コールされた回数がカウントされる。そして，リターン。

```
0032          ENTRY INCHEG(K)
0033           WRITE(6,300)  K
0034      300 FORMAT(1H0,1X,10H***MUSI***,
            $    5X,7HINCHEG(,I6,8H) O MUSI
            $    ,9H SIMASITA)
0035          RETURN
```

6. 以下，INCHEGというサブルーチン。

7. 無視される。ただし，その旨を出力する。そしてリターン。

```
0036          ENTRY ROT(S)
0037           WRITE(6,320)  S
0038      320 FORMAT(1H0,1X,10H***MUSI***,
            $    5X,4HROT(,F12.6,8H) O MUSI
            $    ,9H SIMASITA)
0039          RETURN
0040          END
```

8. 以下，RŌTというサブルーチン。

9. 無視される。ただし，その旨を出力する。そして，リターン。

図9（その5） DS－2のプログラム

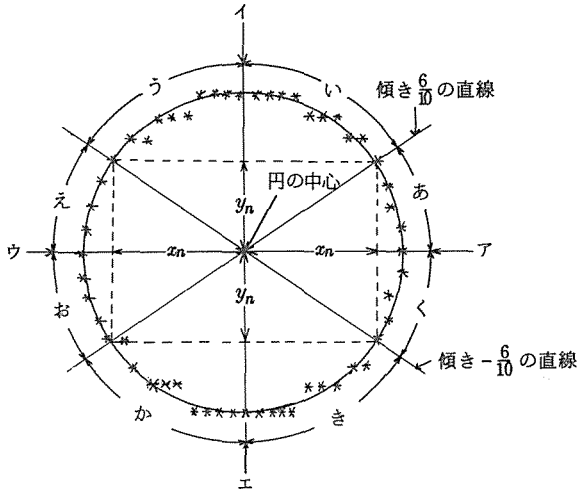
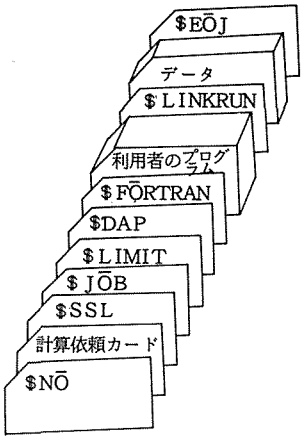


図10. 円（CIRC1）のラインプリンターでの描画

このDAPシミュレーターを使って，現在ドラフターで図を描くプログラムを持っている人が，ラインプリンターで描画する場合は，カードデックを図－11(b)のように構成します。

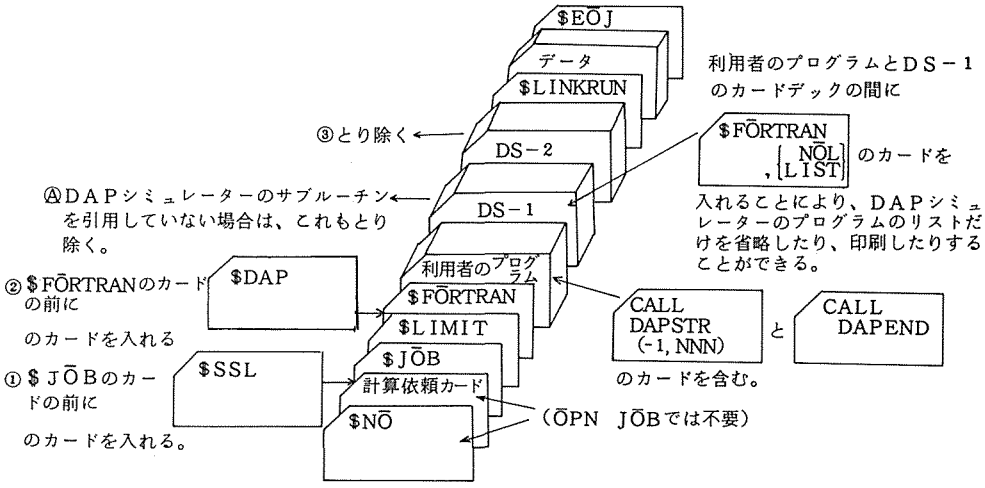
この場合図が大きければ，仮想原点をプリンター原点とするプリント用紙1頁大の図の部分しか描けませんが，図の他の部分を描きたいときは，サブルーチンKIZYUN（X, Y）をメインプログラムの実行文の最初または，CALL　DAPSTR（NŌ，N）の直後に入れます。(図6参照）

大量のグラフをラインプリンターで描く場合は1枚のグラフを描くごとに，PRINT(1)をコールします。それにより，各々のグラフが，それぞれ新しいプリント用紙にプリントアウトされます。また，同じプログラムでドラフターを用いると，一枚の大きな紙に清書されたグラフが自動的に割りつけられます。（図5参照）



(a) ドラフター描画時の標準カードデック構成

（ラインプリンターで描画するときは、$SSL, $DAPのカードを除き、DS−1, DS−2のカードラックを利用者のプログラムの後に入れて、下図(b)のようにする。）



(b) ラインプリンター描画時のカードデック構成

（標準カードデック構成を中央に示す、右にはその補足説明を示す。ドラフターで描く場合には、左の①〜③のようにすればよい。また、DAPシミュレーターのサブルーチンを引用していない場合には、①〜③とⒶのようにすればよい。）

図11. DAPシミュレーターシステムのカードデック構成

（ドラフターで描く場合はBジョブにする必要がある。コントロールカード等については、文献(5)参照。）

```
==DAPSTR==      (-1,222)
==GENTEN==      (    50.000000,    25.000000)
==PEN==       5      (-)
        (      .0       ,      .0       ),(  125.000000,      .0       )
==PFN==       6      (I)
        (      .0       ,      .0       ),(      .0       ,   75.000000)
==PEN==       4      (.)
        (   12.500000,   53.312065),(   15.000000,   53.145668)
        (   15.000000,   53.145668),(   27.500000,   50.671400)
        (   27.500000,   50.671400),(   41.250000,   49.518134)
        (   41.250000,   49.518134),(   50.000000,   48.090361)
        (   50.000000,   48.090361),(   60.000000,   44.918001)
        (   60.000000,   44.918001),(   72.500000,   45.160626)
        (   72.500000,   45.160626),(   81.250000,   40.614643)
        (   81.250000,   40.614643),(   87.500000,   40.011418)
```

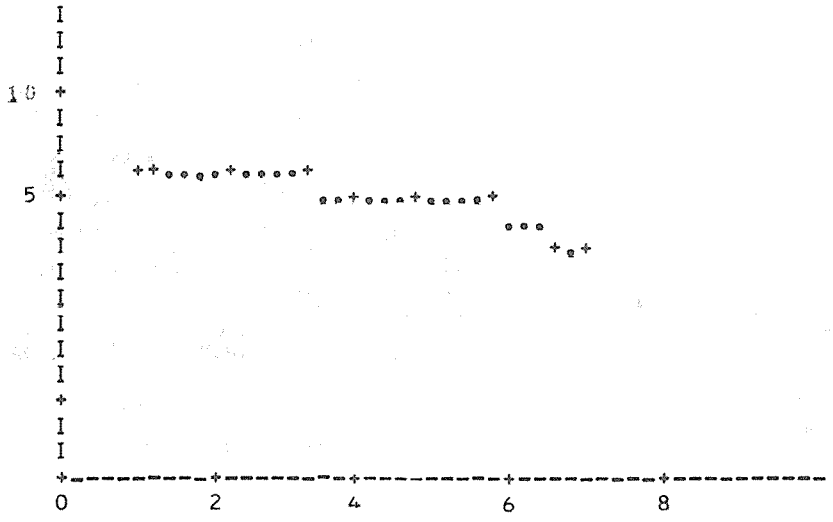


図 13.　実験データのグラフ（ラインプリンター出力の一例）

（上の図は出力されたコメントである。
下の図の横軸は，光の伝搬距離(mm)で縦軸は，光の相対電力（任意単位）の対数をとったものである。
「＋」は測定値を示し，「・・・」でそれらを結んである。）

6. 使用例１：実験データのグラフ化

図１２は，ＤＡＰシミュレーターを利用してグラフを描く簡単なプログラムの一例で，図13と図１４はラインプリンターとドラフターによる出力の一例です。ラインプリンターで描く場合は，まず目にあとから入れた文字の方がプリントアウトされるので，測定データを図上に印すためには，まずＬＩＮＥ１をコールして折れ線の方を画面の配列に入れた後，ＭＡＲＫを用いて測定データを示す「＋」を画面の配列に入れてやる必要があります。

同じような図を４枚ラインプリンターで描き，すべてのプログラムリストを印刷した場合で，ＣＰＵ時間は約１０秒，記憶容量は約６０kch.（約７７００ワード）でした。

なお，ドラフターでは，ＡＰＥＮ（５，-１），ＡＰＥＮ（６，-１）に対して，ドラフター操作卓により，タレット１のみに限定して単色で描いてあります。

```
ISN     LABEL FORTRAN STATEMENT

        C     EXAMPLE(GRAPH-1)
        C       L(NM),VS,LOG(P)
0001            DIMENSION X(20),Y(20),M(5)
0002            COMMON /GC/NGC
0003            DATA M/1H0,1H2,1H4,1H6,1H8/
0004            NGC=1
0005            CALL DAPSTR(-1,222)
0006            CALL GORISA(200.0,150.0)
0007            CALL  ENTER(50.0,25.0)
0008            FX=12.5
0009            FY=50.0

0010      100   READ(5,120,END=500) N,(X(I)
             $    ,Y(I),I=1,N)
0011      120 FORMAT(I6,12F6.0/(6X,12F6.0))
0012            IF(N.LE.0.OR.N.GT.20)
             $    GO TO 500
        C     ZAHYO-ZIKU,MEMORI
0013            CALL APEN(5,-1)
0014            CALL LINE1(0.0,0.0,10.0*FX,
             $    0.0)
0015            CALL APEN(6,-1)
0016            CALL LINE1(0.0,0.0,0.0,
             $    1.5*FY)
0017            DO 130 K=1,5
0018            X1=FLOAT(K-1)*2)*FX
0019            CALL MARK(X1,0.0,1H+)
0020      130   CALL MARK(X1,-5.0,M(K))
0021            CALL MARK(0.0,FY*0.25,1H+)
0022            CALL MARK(-4.0,FY*0.25,1H1)
0023            Y1=FY*(ALOG10(5.0)+0.25)
0024            CALL MARK(0.0,Y1,1H+)
0025            CALL MARK(-4.0,Y1,1H5)
0026            Y2=FY*1.25
0027            CALL MARK(0.0,Y2,1H+)
0028            CALL MARK(-4.0,Y2,1H0)
0029            CALL MARK(-7.0,Y2,1H1)

0030            DO 150 I=1,N
0031            X(I)=FX*X(I)
0032      150   Y(I)=FY*(ALOG10(Y(I))+0.25)
0033            CALL APEN(4,-1)
0034            DO 200 I=2,N
0035            CALL LINE1(X(I-1),Y(I-1),X(I)
             $    ,Y(I))
0036      200   CALL MARK(X(I-1),Y(I-1),1H+)
0037            CALL MARK(X(N),Y(N),1H+)
0038            CALL PRINT(1)
0039            GO TO 100
0040      500   CALL DAPEND
0041            STOP
0042            END
```

0. X，Yはデータを，Mは文字を入れる配列。
   ＮＧＣはコメント制御変数で１（すべてのコメント出力）に設定する。

1. ＤＡＰＳＴＲをコールする。

2. 区画の大きさを設定し，また，原点を移動する。

3. データ変換用の倍率を設定する。

4. データを読み込む。データの個数Ｎが０以下か，２０より大きいときは，１１へとぶ。

5. Ｘ軸を描く，ラインプリンターでは，記号として「－」を用いる。

6. Ｙ軸を描く。ラインプリンターでは，記号として「Ｉ」を用いる。

7. 目盛と数値を描く。ここでは，プログラムを簡単にするため，サブルーチンＭＡＲＫを用いて，「＋」をプロットすることにより，目盛をあらわしている。

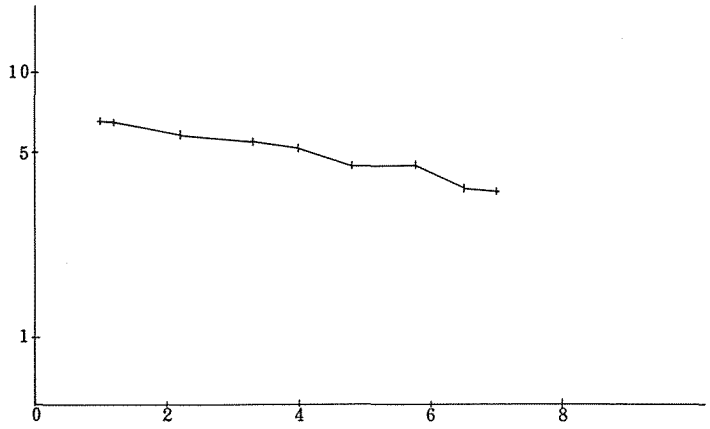
8. データを変換する。（Ｘ座標は拡大，Ｙ座標は，対数をとって平行移動を行ない拡大している。）

9. データを直線（ラインプリンターでは，記号として「＊」を用いる）で結び，各点には，「＋」をプロットする。

10. ＰＲＩＮＴ（１）をコールして，4.へとぶ。

11. ＤＡＰＥＮＤをコールし，ストップ。

図１２．実験データのグラフ化のメインプログラム



NO.　222

図１4.　実験データのグラフ（ドラフター出力の一例）

（横軸は光の伝搬距離（mm）で、縦軸は光の相対電力（任意単位）の対数をとったものである。
測定値の位置を「＋」で示し、それらを折れ線で結んである。）

7.　使用例２：ステレオ透視図

立体の透視図を描く場合は，画面と視点，立体の各点の位置がわかれば，画面と視線との交点の集合として描くことができます。（たとえば，文献（１）６０ページ参照）そして，図15のように，右目と左目のおのおのに対応する透視図を描いてやれば，図１６のような装置で，対象を立体的に見ることが出来ます。

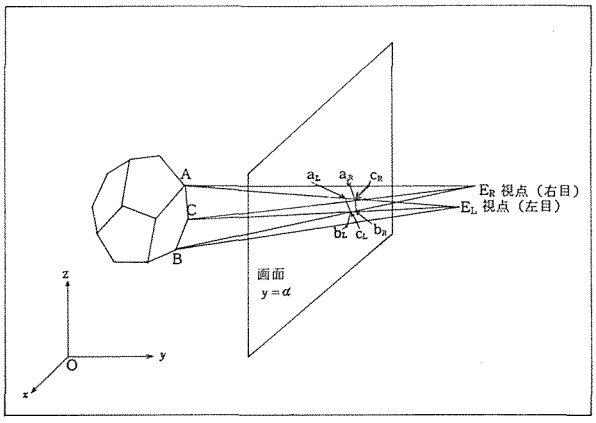


図１5.　ステレオ透視図の考え方

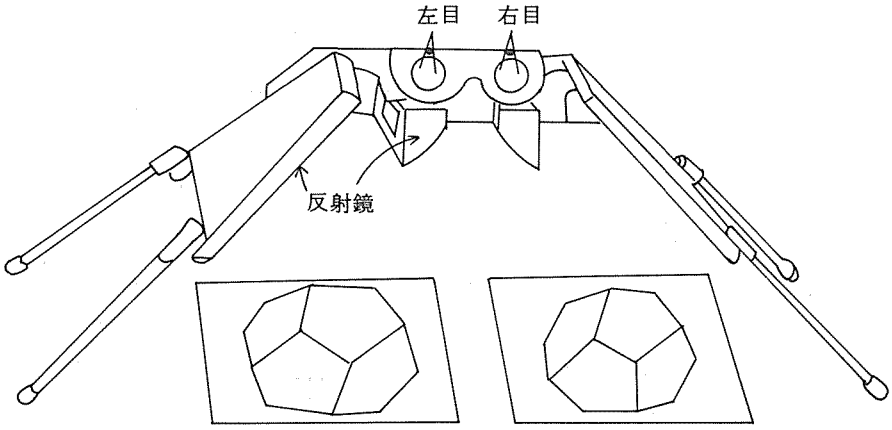


図16. 反射実体鏡

図17と図18は，正十二面体を左右の目で見たステレオ透視図です。ここでは，隠れ線は，P. P. Loutrel[6]の方法を用いて判定し，破線で描いてあります。破線を構成する各線分の長さは，3次元空間で等長となるように計算して描画してありますが，これは従来，手作業による作図では，ほとんど不可能に近い作業でした。ラインプリンターで描く場合は，解像力の関係で，この破線はうまく描けないため，「＊」のかわりに「＋」を用いてこれをあらわしてあります。

ＰＲＩＮＴ（Ｎ）をうまく使えば，赤と緑で，左右の目に対応する透視図を描くことも可能です。

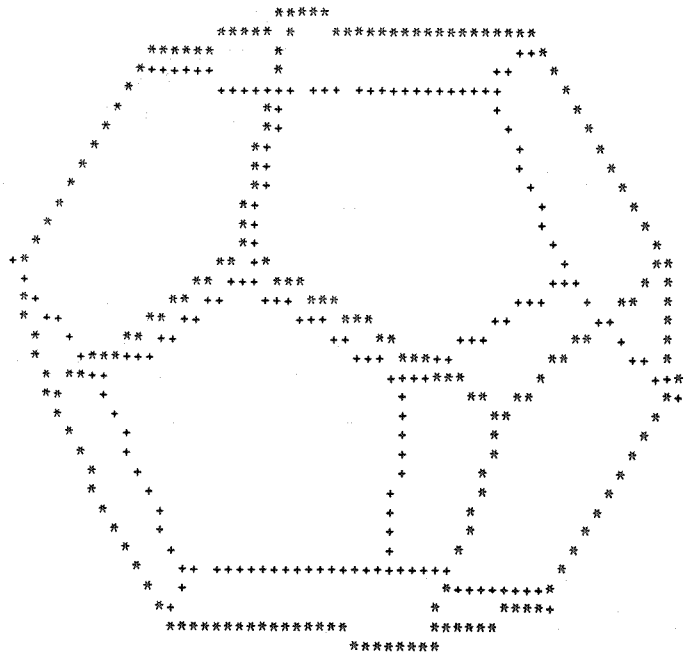


図１７（その１）　ラインプリンターによる正十二面体の透視図（左目用）

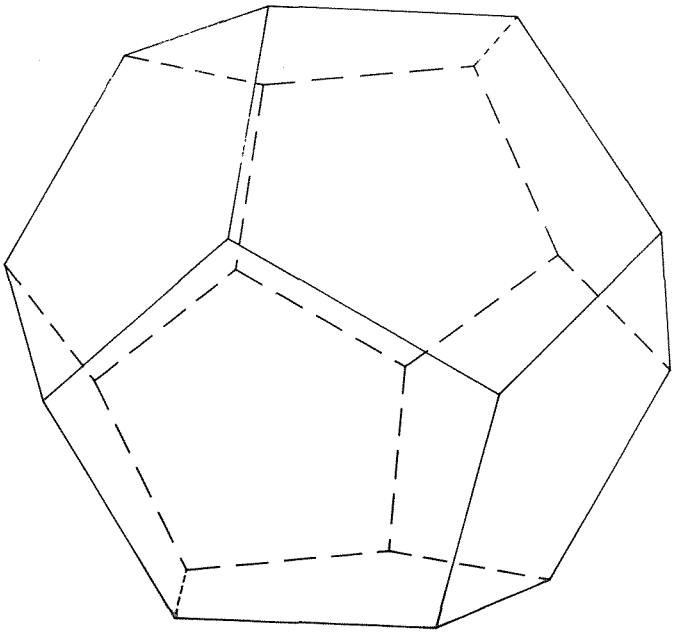


図１８（その１）　正十二面体の透視図（左目用）

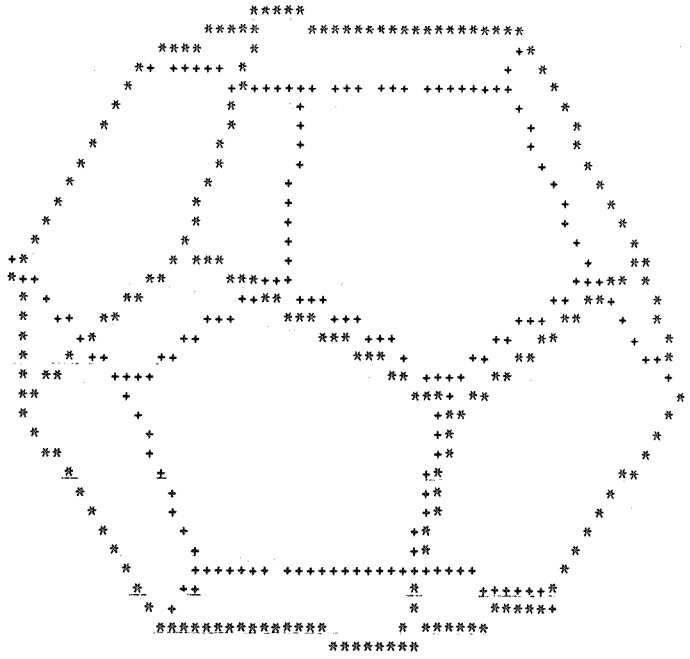


図１７（その２）　ラインプリンターによる正十二面体の透視図（右目用）

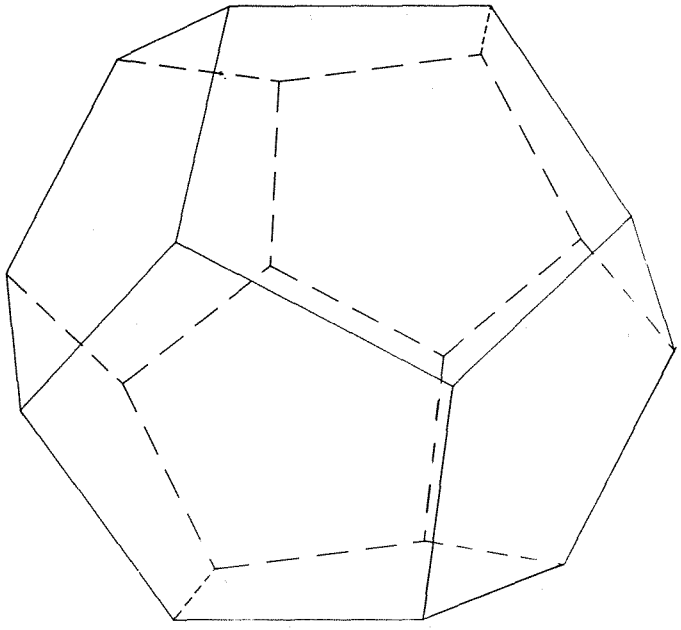


図１８（その２）　正十二面体の透視図（右目用）

8. 考察と結論

使用例1は，約５０ステートメントのプログラムですが　ＤＡＰシミュレーターシステムを用いて，オープンバッチジョブで３回流すことにより，図１３のような結果を得，ドラフターで間違いなく描けるとの確信を得ることができました。そして，その日のうちにドラフタージョブとしてＢジョブに流すことができました。オープンバッチジョブによるディバッグがなければ，おそらくこのような確信を得るまでに約１週間はかかるところです。このように，ＤＡＰシミュレーターシステムを用いることにより，プログラムを流し始めてから，目的の結果を得るまでの実質的なターンアラウンドタイムは，大幅に短縮できました。

ビット関数は，当初，オープンバッチジョブにかけられることを目的として導入したのですが，ラインプリンターによる描画の場合，プログラムによっては，実行時間の大幅な短縮になることがわかりました。

表６は，５６×１３０の画面のます目を，ビット関数を用いず配列ＮＧＡ（５６，１３０）とした場合とビット関数を用いて配列ＮＧ（５６，１７）とした場合について，すべてのます目にクリア，代入，印刷を実行した場合のおおよそのＣＰＵ時間です。

表6.　ビット関数を用いた場合と用いない場合のＣＰＵ時間の比較＊1

	ビット関数使用せず ＮＧＡ（５６，１３０）	ビット関数使用＊2 ＮＧ（５６，１７）
クリア	約　４８ msec	約　９ msec
代入	約　４８ msec	約　８５０ msec
印刷	約　１１２０ msec	約　１７５ msec

＊1　これらのＣＰＵ時間はいずれも５６×１３０のます目について，２重ＤＯループを用いて行なった結果である。

＊2　実際にビット関数を用いているのは，代入のときだけである。他については５６×１７の各配列要素に対して行なっている。

ふつう，グラフや図をかくようなプログラムでは，クリア，印刷は，必ず全ます目に対して行なわれるのに対し，代入は，一部のます目に対して行なわれるだけです。すなわち，５６×１３０＝７２８０回行われるわけではなく，普通は，１０００回以下です。したがって，代入に要するＣＰＵ時間は，１／７以下になることが多く，トータルではビット関数を用いない場

合で約１１７５ msec，ビット関数を用いた場合では約３０５ msec，となり，ＣＰＵ時間は，約１／３となるわけです。

このシステムは，阪大センターのＤＡＰシステムを基礎として開発したものですが，簡単な変更を加えれば，他の計算機システムのラインプリンターによる描画システムとすることも可能です。その場合の主な変更点と注意点は次の３点です。

その１は，Entry文のサブルーチン化です。その場合，Common 文，Data 文については，図９に示すように，Entry ごとに，注釈行として，Common 文についてはブロック名を，Data 文については変数名または配列名を入れてありますので，それを手がかりにつけ加える必要があります。

その２はビット関数の処置です。他のシステムで用いる場合は，それぞれの計算機に合うように，サブルーチンＮＴＲ（Ｉ，Ｊ，ＫＩＧＯＮ）に相当するサブルーチンを作成する必要があります。それが困難な場合には，記憶容量，計算時間は増加しますが，

ＣＯＭＭＯＮ／ＧＮ／ＮＧ（５６，１７）は，ＣＯＭＭＯＮ／ＧＮ／ＮＧＡ（５６，１３０）に，ＣＡＬＬ　ＮＴＲ（Ｉ，Ｊ，ＫＩＧＯＮ）は，ＮＧＡ（Ｉ，Ｊ）＝ＫＩＧＯＮとして，配列のクリア，出力の部分を変更すれば使えます。

その３は，Data 文が使えない場合で，Read 文あるいは適当な方法で初期値を設定します。

ところで，本ＤＡＰシミュレーターでは，オープンバッチジョブが使えるということが，システム設計の優先条件となっていますので，ＤＳ－２に入っているサブルーチンは，ＤＡＰシステムのサブルーチンのうち，ごく基本的なもののみです。しかし，それ以外のサブルーチンについても，ラインプリンター描画用のサブルーチンを作り出すことは可能です。

それらを作り出すのに次の２つの方法があります。

その１は，ラインプリンター向きのプログラムを新たに作り出す方法です。

その２は，基本的なＰＬＯＴ，ＡＲＣ１，座標系変換ルーチン等についてラインプリンター用のものを用意し，ＤＡＰシステムのＤＡＰＳＴＲをコールしてやる方法です。この場合，たとえば，ＣＩＲＣ１を利用者のメインプログラムでコールすると，ＤＡＰシステムのCIRC１が呼び出され，その中でＡＲＣ１が引用されるとき，ＤＡＰシステムのＡＲＣ１にかわって，ラインプリンター描画用のＡＲＣ１が実行されるわけです。

この２つの方法は，いずれも問題があります。たとえばＳＹＭＢＬ１，ＳＹＭＢＬ２，ＮＵＭＢＲ等については，第１法としては，ＭＡＲＫと同様のプログラムを作り出すことが可能ですが，文字の大きさ，傾きは犠牲になります。第２法は，ＤＡＰシステム内のサブルーチンを利用する方法ですが，文字の大きさが小さいと，描かれた文字が識別できなくなります。

ＡＲＲＯＷについては，第１法としては，図１９右のような図をかいて矢印とみなす方法があります。しかし，できた図が，一般の人に矢印であると説明なしに認識させるのはむずかしいと思われます。

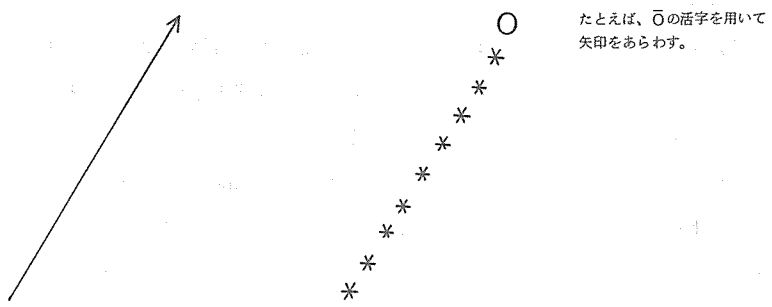


図19.　ドラフターとラインプリンターによる矢印

第２法としては，上記の場合と同様に，ＤＡＰシステムのＡＲＲＯ̄Ｗを利用する方法があります。しかし，ラインプリンターによる図の解像力の関係で，矢印は出てこないことが多いと思われます。

なおＤＡＰシステムのサブルーチンを第２法のように利用しようとすると，Ｂジョブでしか流せなくなるのも，問題点の１つです。

以上，筆者らの開発したＤＡＰシミュレーターシステムにまつわることがらについて，記しました。こうしたシステムを開発することにより，その副次的な効果として，ＤＡＰシステムそのものへの理解を大幅に深めると共に，より使いやすいＤＡＰシステムを作るための手がかりを見出すことも出来ました。

その１は，ＤＡＰシステムそのものと，ＤＡＰシミュレーターを一貫して作ることにより，もっと無理のない，きめの細かいプログラムが組め，ラインプリンターによるより使いやすく，効率的なシミュレーターを含むＤＡＰシステムが作れるということです。

すでに，述べたように，ＰＬＯ̄Ｔ，ＡＲＣ１等以外の基本的でないサブルーチンは，ドラフター，ラインプリンターで共用できますし，一貫して作ることにより，これらの倍率の変化，原点の移動が，ラインプリンター描画でも，ドラフター描画でもトレースでき，ディバック時に，偉力を発揮させることが可能となるからです。

その２は，ここで作成したサブルーチンＰＲＩＮＴ（Ｎ）に相当する機能をもつサブルーチンは，こうした一貫したシステムには，不可欠であるということです。このサブルーチンにより，ラインプリンターで大量にプリントアウトした図の中から任意のものを任意の枚数だけ手軽にドラフターに清書させることが可能となる等，研究者の使い方にマッチしたシステムとすることが可能となるからです。

今後，実際に研究実務の上でこのシステムを生かして行くことにより，なお不十分な点などを整理していく予定です。

本研究には，大阪大学大型計算機センターのＮＥＡＣシリーズ２２００（７００）とＤＡＰシステムを用いました。ドラフターの掛の方には，色々とドラフターの誤動作等について教えをいただきましたが，ここに謝意を表します。

このＤＡＰシミュレーターシステムが，ドラフター使用についての拒絶反応を，いくらかでも柔らげる効果があれば幸いです。

9. 文　献

(1) 吉田勝行（１９７３）「ラインプリンターとディジタル・グラフィクス」大阪大学計算機センター・ニュース，№１０，ＰＰ　５３～７０

(2) 大野義夫（１９７５）「ラインプリンタによるカーブプロッタのシミュレータ」情報処理，Vol.１６，№１１，ＰＰ．１０２８～１０３０

(3) 日本電気KK，（１９７３）「ＮＥＡＣ－シリーズ２２００，オペレーティングシステムＭＯＤⅣ ＥＸ／Ⅶ ＦＯＲＴＲＡＮ７００文法説明書」

(4) 大阪大学大型計算機センター（１９７５）「自動製図装置（ドラフター）使用説明書」

(5) 大阪大学大型計算機センター（１９７５）「利用の手引」

(6) ＰＨＩＬＩＰＰＥ　Ｐ．ＬＯＵＴＲＥＬ（１９７０）「A Solution to the Hidden－Line Problem for Computer－Drawn Polyhedra」PP. ２０５～２１３

付　　録

筆者が，ドラフターを用いて描画した際，気づいた阪大センターのドラフターの製図用サブルーチンの問題点を以下に述べる。

1. 以前のことであるが，

   ＰＬＯＴ　（Ｘ，Ｙ，－２）

   ＰＬＯＴ　（Ｘ，Ｙ，－３）

を用いて原点移動を行なう場合，原点の移動が正常に行なわれなかった。これに対し，阪大センターは，その後ＰＬＯＴ（Ｘ，Ｙ，－２），ＰＬＯＴ（Ｘ，Ｙ，－３）を使用説明書から省いた。

2. ＭＡＲＫ（Ｘ，Ｙ，ＩＢＣＤ）は一時，正常に動作しなかったが，現在は正常である。

3. ＮＯＳＥＱとＧＥＮＴＥＮの両方のサブルーチンを引用すると，原点が正常に移動しない。現在も改善されていない。

4. ＩＮＣＨＥＧ（Ｋ）について新しい自動製図装置（ドラフター）使用説明書（文献(4)）では，Ｋ＝２とＫ＝３が入れかわっている。これは間違いで古い説明書の方が正しい。

5. ＧＥＮＴＥＮとＩＮＣＨＥＧの両方のサブルーチンを引用すると，正しく描けない。

6. ＳＭＯ̄ＴＨは，ほとんど，折れ線で結んだのと変わらない。また，閉じた曲線を描いたところ，正しく描画されなかった。

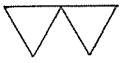
7. ＳＨＡＤＥは，文献（４）３４ページの例３のように閉じた領域において行なったにもかかわらず，斜線の傾きによっては，正しく描けない場合があった。

8. ＳＬＩＮＥ（Ｘ，Ｙ，Ｎ，Ｋ，ＩＴＹＰＥ，ＩＣＯ̄ＤＥ）で，ＩＣＯ̄ＤＥに４６を用いたが，つけられたマークは正しくなく０００００００と１点に０が７個ずつ描かれた。

9. ＳＹＭＢＬ１（ＸＳ，ＹＳ，Ｈ，ＩＣＯ̄ＤＥ，Ｓ）で描かれる記号のうち，ＩＣＯ̄ＤＥ６３，６４に対応する記号は，表７のように２字分，３字分の幅が用いられて描かれる。

以上が，気づいた問題点である。一部，筆者の思い違いがあるかも知れないが，大部分は，システムのサブルーチンの不備のようであり，早期の改善が望まれる。

表７．サブルーチンＳＹＭＢＬ１の問題点＊1

ICODE	現在、ドラフターで描かれる図	本来、描かれるべき図
６３		
６４		
６５		

＊１　ＩＣＯ̄ＤＥ６３に対する図は正しいが、ＩＣＯ̄ＤＥ６４，６５に対しては、本来１文字分で描かれるべきであるのにそれぞれ２文字、３文字分用いて描かれる。

御指摘の問題点について

センター

御指摘のあったＤＡＰ－Ｏ・サブルーチン群の問題点については、以前からセンターでも気付いていることです。ＤＡＰ－Ｏは、武藤工業提供のソフトウェアーを、ＮＥＡＣ 2200 シリーズ・モデル700 用として、コンパイル時にエラーが出ないよう変更し、ライブラリーに登録したものであり、プログラムのアルゴリズムについては、変更しませんでした。ところが、これをユーザーに開放したところ、続々と問題点が発生し、これらについては、その都度、原因を調査し、わかったものについてはプログラムの修正を行なってまいりました（例えば、ＥＬＩＰＳ，ＤＩＭＡＮ，ＭＡＲＫ等）。しかし、利用者プログラムは、複雑さを増す一方であり、それにともなってＤＡＰ－Ｏのプログラムの問題点も増え、理想的な作図のできない事態も発生してまいりました。御指摘のあったものは、そのサブルーチンのアルゴリズム自体に問題があり、修正するとすればまったく新しいプログラムを作成しなければならないもの（ＳＭＯＴＨ等）が多く相当の作業量となります。

これらについては、随時センター側でも原因を調査し、修正していく予定ですが、利用者におかれましても、しばらくの間、問題点のあるサブルーチン（ドラフター室にも掲示）については、使用しないようにお願いします。