3次元図形記述言語 GRADELATHIN†

上 内 秀 隆††

Abstract

It is the well-known fact that is very difficult to eliminate three-dimensional hidden-lines using any existing methods. This is one of reasons why it is difficult to process three-dimensional informations. We have concluded that it is very useful or effective that we use three-dimensional triangles as elementary surfaces of three-dimensional informations and obtain several algorithms to compute outlines, intersections and hidden-lines as for surfaces which can be approximated with triangles. Thus we designed a graphic description language for three-dimensional informations GRADELATHIN based on our conclusion and described its preprocessor in this paper.

1. はじめに

3次元図形の hidden-line 消去を一般的な手法で行なうのは，大変に難かしく，このことが，3次元図形の一般的処理を困難にしている理由の一つとなっている．hidden-line 消去法は，図形情報の性質によって，強い制約を受けるので，一般図形に対する一般的なアルゴリズムを作るのは，難かしい．しかしながら，一般図形情報を，処理システム固有の図形情報に近似的に変換することで，対象は限られるが systematic な hidden-line 消去が可能となる．われわれも図形が，3角形の集合で近似できる曲面については，輪郭線，相貫線の検出[5]，および，hidden-line 消去[4,6]が完全にできることを知っている．このような3つの論文の結果を基礎にして，考察を進めたとき，3角形を3次元図形の基本曲面 (Elementary Surface) 素として使用することは，有意義であるとの結論を得た．この結論に基づき，3次元図形記述言語 GRADELATHIN を設計し，そのプリプロセッサーを試作してみた．曲面設計上，よく使用する patch 曲面[7,12]なども，記述の対象に含めてみた．本文では，記述言語の設計思想シンタックス，およびそのプリプロセッサーについて議論する．

† A GRAPHIC DESCRIPTION LANGUAGE FOR THREE-DIMENSIONAL INFORMATIONS—GRADELATHIN, by Hidetaka KAMIUCHI (Kyoto Unversity, in present Electrotechnical Laboratory)

†† 京都大学工学部数理工学教室，(現在) 電子技術総合研究所パターン情報部図形処理研究室

2. 基本曲面素の決定

2.1 基本曲面を必要とする理由

3次元図形記述言語には，次の様な特徴が要請される．すなわち，

(A_1) 多様な3次元図形の記述が可能であること．すなわち，特定の凸体に限定しない任意の図形の記述が可能であること．

(A_2) 図形記述と，任意の図形計算との連結が可能で，かつ，論理的矛盾を生ぜしめないこと．

しかしながら，すべての3次元図形を網羅したり，すべての，あらゆる場合を想定して，図形記述言語を作るのは，不可能に近い．したがって，表現可能な図形対象を取捨選択することが必要である．そこで，簡単な言語構造を持ち，かつ，多様な表現が可能である3次元図形記述言語を考えることにすると，すぐに，種々の計算処理が可能で，かつ簡単な構造の図形要素を設定し，これを構成基本要素として，多様で，かつ複雑な図形を構成的に記述する方法を思いつくであろう．しかし，この方法の具体化のためには，何を基本図形に選ぶかという基本的な問題を解決しておく必要が生じる．これに対し，われわれは，次のような判定基準を考える．

(B_1) 単純な3次元構造を持つこと．

(B_2) 直感的に判り易く，取り扱い易い図形であること．

(B_3) 複雑な曲面の近似的構成が，理論的にも，実際的にも，可能であること．

(B4) 基本的図形処理において，収束型計算をさけ得ること．

われわれは，3次元図形の hidden-line 消去，輪郭線，相貫線について，詳細に調べたが，そこでの結論は次のようなものであった．

(C1) hidden-line 検出法に関する結論[4),6)]．

任意の曲面が，障害物として，存在する場合には，hidden-line を検出する計算は，条件付非線型方程式の解の存在を調べる問題に帰着される．したがって一般的に言えば，収束型計算を必要とする．収束型計算を用いずに，2次式，または，完全因数分解型の連立2次不等式の範囲内で解ける曲面は，次のようなものであった．

$$Q(x,y,z)=a_1x+a_2y+a_3z+a_4=0, \qquad (2\text{-}1)$$

$$Q(x,y,z)=a_1x^2+a_2y^2+a_3z^2+a_4xy+a_5zy+a_6zx+a_7x+a_8y+a_9z+a_{10}=0, \qquad (2\text{-}2)$$

$$\boldsymbol{R}=\boldsymbol{A}t+\boldsymbol{R}u+\boldsymbol{C}, \quad 0\leq u,\ t\leq 1 \qquad (2\text{-}3)$$

$$\boldsymbol{R}=\boldsymbol{A}wt+\boldsymbol{B}t+\boldsymbol{C}, \quad 0\leq w,\ t\leq 1 \qquad (2\text{-}4)$$

$$\boldsymbol{R}=\boldsymbol{A}ut+\boldsymbol{B}u+\boldsymbol{C}t+\boldsymbol{D}, \quad 0\leq u,\ t\leq 1 \qquad (2\text{-}5)$$

$$\boldsymbol{R}=\boldsymbol{A}uvt+\boldsymbol{B}vt+\boldsymbol{C}t+\boldsymbol{D}, \quad 0\leq u,v,t=\leq 1 \qquad (2\text{-}6)$$

である．各式は，直接変数表示の平面，直接変数表示の2次曲面，間接変数表示の平行四辺形，間接変数表示の3次元3角形，間接変数表示の特殊な2次曲面，間接変数表示の3角錐である．この他に，間接変数表示の四角錐がある．

(C2) 輪郭線の検出に関する結論[5)]．

輪郭線を求めるには，次のような操作が必要である．対象とする曲面上のある点における接平面を考えこの接平面が，視線を含めば，その点は，可輪郭線の一部である．すなわち，ある曲面が与えられたとき，曲面とその接平面と視線の連立方程式の縮退として，可輪郭線が求まる．その縮退した方程式を満す解がすべて，輪郭線となるので，一般には，収束型計算を必要とする．もちろん，特異点も，可輪郭線の一部となっている．2次曲面を与えた場合，接平面が線型になるので，連立方程式は線型となる．しかし，可輪郭線を追随しようとすると，その計算は，面倒なものになる．ところが，3次元3角形の集合に関しては，論文5）の定理2より，その頂点の座標を用いて，簡単な計算で可輪郭線が求まることが判った．

(C3) 相貫曲線 (intersection) の検出法に関する結論[5)]．

ある曲面とある曲面の相貫曲線は，それらを表現する方程式を連立させたものの解として求まる．よって一般には，非線型連立方程式となるので，収束型計算が必要であり，しかも，かなり，やっかいである．ところが，3角形の集合に関しては，論文5) の計算法に従がえば，3角形間の相貫を計算することができる．そこでは，収束型の計算は，必要ではない．

(C4) 3次元3角形の汎用性について[4),5)]．

3次元曲面の3角形近似は，常に可能である．実際に，近似を実現するのは，それほど困難ではない．また，近似精度の評価が可能である．

以上の結論 C1～C4 からも判るように，3次元曲面の基本要素としては，3次元3角形，すなわち，式 (2-4) で表現されるものが適当であるとの結論を得た．しかし，当然のことながら，patch 曲面等のよく使用する非線型の曲面の取り扱いをどのようにするかという疑問が生じる．これらの非線型曲面は，hidden-line, 輪郭線，相貫曲線などの計算が困難であるという点で，図形処理において，弱点を有するものであった．しかし，patch 曲面などの非線型曲面は，図形処理時に，3角形の集合で近似すれば，困難は，避けられることが判った．こういう意味からも，3次元3角形を基本面とすることに決定するものである．

2.2 データ構造からの規点

データの構造とは，単なるデータの配列，そのものではなく，その配列の規則性，構造性を示すところの概念「データ」に対する上位概念である．コンピュータ・グラフィックスにおけるデータ構造を議論する場合，次のような階層構造を考え，そのうちのどれに的を絞るかを，あらかじめ明白にする必要があろう．通常コンピュータ・グラフィックスで，データ構造と言う言葉が用いられるのは，次のような意味においてである．

(i) 第1種データ構造

CRT ディスプレー装置では，これを駆動するのに小型のコンピュータを必要とする．このとき，制御命令，文字列，座標列，入力識別用 area などの，データ列が，メモリーの中にたくわえられる．それが，常時，走査される．このように，CRT ディスプレー装置などに密着したデータ配列の型を第一種データ構造と呼ぶ．プロッターなどを駆動する場合には，極めて簡単な構造，すなわち，座標列のみである．

(ii) 第2種データ構造

Graphic Subroutine Package (GSP)[11)] におけるデータ構造を指す．グラフィック・データは，いくつ

かのグラフィック・ファイルからなり，グラフィック・ファイルは，グラフィック・エレメントからなる．このエレメントは，座標列，文字列，制御語列などからなり，ディスプレー出力，識別などに関する単位を設定する．2次元図形を要素として，ファイルを構成するという点で，2次元図形用データ構造を持つ．

(iii) 第3種データ構造

一般図形の関係，および構成に関するデータ構造である．3次元物体の構成などが，これに相当する．第2種データ構造の上位概念にあたる．これは，人間の物に対する概念に近い構造を有するもので，図形，物体の表現に関する抽象的なデータを対象とする．

われわれが，GRADELATHIN によって，記述しようとするデータ構造は，第3種に属するものである．したがって，記述形式は，通常，われわれの用いる思考形式に近く，かつ，使用する概念も，通常の概念に近いものであることが望ましい．また，第3種のデータ構造を持つ．言語によって，図形情報が与えられたとき，いろいろな操作（ディスプレー，データ変換，修飾，データ変更など）が可能であることが望ましい．通常，第1，第2種のデータ構造は，システム側に，完備されているものであった．しかし，それは，ユーザのプログラム中に，implicit に記述されるものである．われわれの GRADFLATHIN は図形情報に関するなるべく多くの部分を，explicit に表現して，図形の取り扱いをなるべく容易にしようと考えるものである．このため，第3種のデータ構造は，GRADELATHIN によって規定されるものとなる．

3. 3次元図形記述言語 GRADELATHIN の構成

3.1 2 mode 方式による構成

3次元記述言語は，2つの mode すなわち，DECLATION MODE と PROCEDURE MODE とで構成される (Fig. 1)．DECLARATION MODE は DECLARATION MODE ステートメントからなり，3次元図形を記述する．そして，各ステートメントが続き，END ステートメントで終る．PROCEDURE MODE は，すでに記述された DECLARATION MODE の図形記述を用いて，種々の情報処理を行なう．この MODE は PROCEDURE MODE ステートメントで始まり，FORTAN-like ステートメントの集合が続き，PROCEDURE END ステートメントで終る．DECLARATION MODE は PROCEDURE

```
DECLARATION MODE
  [ Graphic Description
END
PROCEDURE MODE
  [ Fortran-like Procedure
PROCEDURE END
```

Fig. 1 2-mode-type construction

MODE に先立つ必要がある．このようにして，構成した，われわれの3次元記述言語を GRADELATHIN と呼ぶ．これは，a GRAphic DEscription LAnguage for THreedimensional INformations の略称である．

3.2 DECLARATION MODE 中の各ステートメント

この MODE では，3次元図形を直接記述するステートメント (No. 1～No. 10) と，プリ・プロセッサー (translator) に対する制御を行う制御用ステートメント (No. 11～No. 15) の2つから構成されている．3次元図形を直接記述するステートメントのシンタックスは，Fig. 3 に示す．各ステートメントの持つ意味は，次のとおりである．

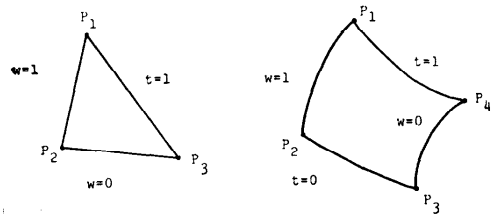
No. 1 SPACE ステートメント；対象とする曲面，視面，視点を定義する．

No. 2 SURFACE ステートメント (OBJECT ステートメント)；(i) ある曲面に対する回転，対称，平行移動，拡大縮小を指示する．指定順を変更してはいけない．(ii) 成分曲面を列記する．

No. 3 EPLANE ステートメント；基本曲面である3角形を定義する．曲面は，

$$\boldsymbol{R}=(\boldsymbol{P}_1-\boldsymbol{P}_3)wt+(\boldsymbol{P}_3-\boldsymbol{P}_2)t+\boldsymbol{P}_2,\quad 0\leq w,\ t\leq 1$$

で表現される．3つの頂点は $(w,t)=(1,1)$，$(w,t)=(\alpha,0)$，$(w,t)=(0,1)$ に対応する．あとに続く3つの整数は，それぞれ $w=1$，$w=0$，$t=1$ なる辺の状態を示すものである．この辺が visible であって，plottable であるとしてよいとき，1，そうでないとき，0 が書かれる．これは3角形で，任意曲面を近似するのに必要なパラメータである．EPLANE の定義方法には，(i) 3点で定義する，(ii) 1線と1点で定義する，などの場合があるが，その時の線は，有限座標を持つ両端を持たねばならない (Fig 2 (a))．



(a) a triangle (b) a quadric surface and a patch surface

Fig. 2 EPLANE and QDSURFACE and PATCH

```
Syntax ;

SPACE-(.)=SURFACE-(.) VIEW-($) VIEW-(::)

SURFACE-(.)=ROTARY-(.)? SIMMETRY-(#)? PARALLEL-(;)?
      EXPANSION-(;)? SURFACE-(.) /
      SURFACE-(.)@ EPLANE-(.)e QDSURFACE-(.)e PATCH-(.)$ -ε

EPLANE-(.)=POINT-(.) POINT-(.) POINT-(.) (*) (*) (*) /
      LINE-(.) POINT-(.) (*) (*) (*)

QDSURFACE-(.)=POINT-(.) POINT-(.) POINT-(.) POINT-(.) (*) (*)
      (*) (*)

PATCH-(.)=PARA-(.)@ -ε  (*) (*) (*) (*)

PARA-(.)=PARX-(:) PARY-(:) PARZ-(:)

POINT-(.)=CODX-(:) CODY-(:) CODZ-(:) / INTERSECT-(LINE-(.)
      EPLANE-(.))

LINE-(.)=POINT-(.) POINT-(.) / POINT-(.) COS-(.) /
      INTERSECT-(EPLANE-(.) EPLANE-(.))

COS-(.)=COSX-(:) COSY-(:) COSZ-(:)

ROTARY-(.)=THETA-(:) FAI-(:) / null

Signatures ;

(.)= variable name / array element
(:)= variable name / array element / real number
(*)= integral number ( 0 or 1 )
(;)= POINT-(.) / null
(#)=POINT-(.) / LINE-(.) / EPLNAM-(.) / null
(::)=EPLANE-(.)
($)=POINT-(.) / COS-(.)

Remarks ;

 ? ;     A? means A or nothing.
 @ ;     A@= ε ∨ A ∨ AA ∨ AAA ∨ AAAA ∨ AAAAA∨.....
-ε ;     this means elimination of ε . ε is an empty symbol.
 OBJECT ; we can use OBJECT- in stead of SURFACE-, and both.
 INTERSECT-; LINE-name must not be defined with INTERSECT-.
       EPLANE-name must have been defined already.
       LINE-name composing EPLANE-name must have the finite
       terminates.
null ; the word NULL is written. This means no operation.
```

Fig. 3 Syntax of DECLARATION MODE

No. 4 QDSURFACE ステートメント；4点で定義される2次曲面である．各点から，

$$\boldsymbol{R}=(\boldsymbol{P}_1+\boldsymbol{P}_3-\boldsymbol{P}_4-\boldsymbol{P}_2)wt+(\boldsymbol{P}_4-\boldsymbol{P}_3)t+(\boldsymbol{P}_2-\boldsymbol{P}_3)w+\boldsymbol{P}_3 \quad 0\leq w,\ t\leq 1$$

と決る．あとに続く4つのパラメータは，0か1をとる整数で，$w=1$, $t=0$, $w=0$, $t=1$ に対応する辺の特性を示す．意味は，No. 3 の場合と同じである (Fig. 2 (b))．

No. 5 PATCH ステートメント；これは，

$$\boldsymbol{R}=\sum \boldsymbol{A}_{ij}w^i t^j, \quad 0\leq w,\ t=1$$

なる patch 曲面を定義する．パラメータ $\boldsymbol{A}_{ij}$ は w, t のべき数の和の小さいものから，また，和が一定のとき，べき数の小さいものから順に対応する．あとに続く，4つのパラメータは，No. 4 と同義に用いる．(付録 1, 2 参照.)

No. 6 PARA ステートメント；3つの変量で，パラメータ・ベクトルを定義する．

No. 7 POINT ステートメント；(ⅰ) 3つの座標変量で定義する．(ⅱ) 線と基本面の交点で定義する．

No. 8 LINE ステートメント；(ⅰ) 2つの点，(ⅰ) 1点と方向余弦，(ⅲ) 基本曲面と基本曲面との交りで定義する．

No. 9 COS ステートメント；3つの変量で定義する．

No. 10 ROTARY ステートメント；2つの角変量（度数，実数）で定義する．

などである．

これらの図形記述用ステートメントの他に DECLARATION MODE の制御用ステートメントとして，

No. 11 DIMENSION ステートメント；曲面の名前などに配列表示を可能にするため，あらかじめ，その宣言を行う．

No. 12 DO ステートメント；すぐ，次の位置するステートメントを DO のあとに続く変量だけ繰り返し，定義する．

No. 13 COMMENT ステートメント；COMMENT であり何も行なわない．DO ステートメントの直後に位置しては，いけない．

No. 14 DECLARATION MODE ステートメント；DECLARATION MODE の始りを，プリプロセッサーに伝える．

No. 15 END ステートメント；DECLATION の終りをプリ・プロセッサーに伝える．

3.3 PROCEDURE MODE の各ステートメント

PROCEDORE MODE では，次のような各ステートメントが許される．

No. 1 PROCEDUSE MODE ステートメント；PROCEDURE MODE の始りを示す．

No. 2 PROCEDURE END ステートメント；PRODEDURE MODE の終りを示す．

No. 3 FORTRAN Ⅳ with GSP に含まれるすべてのステートメント．

No. 4 DISPLAY ステートメント；引数を5つ持つ．すなわち，(ⅰ) SPACE-name, (ⅱ) Graphic File-name, (ⅲ) Graphic Element-name, (ⅳ) Graphic File-size である．後の3つは，整数，または，整変数である．SPACE-name には，DIVISION-name すなわち，SPACE- を付けておく．第5の引数は，整数，または整変数であり，その値が0のとき hidden-line の消去を行い，1のとき，hidden-line を破線で示し，それ以外のとき，hidden-line の消去処理をしない．

なお，この MODE においては，次の変数に限り，DECLARATION MODE との共通使用ができる．他のものは，全く相関を持たない．

(ⅰ) SPACE-name；ただし，DISPLAY ステートメントの中のみ，有効である．

(ⅱ) real variable name；座標，方向余弦，角度およびパラメータに使用した変量は，共通使用可能である．

この (ⅰ) (ⅱ) の name の使用を通じて，2つの MODE の間の連結が行なわれる．

4. GRADELATHINのプリ・プロセッサー

4.1 プリ・プロセッサーの概略

GRADELATHIN で書かれたプログラムが，プリ・プロセッサーの原始プログラムとなる．プリ・プロセッサーは，これらを GSP を持った FORTRAN に変換される．このようにして作られたオブジェクト・プログラムは磁気テープに格納される．この磁気テープのプログラムは，FORTRAN コンパイラーの原始プログラムとなる (Fig. 4)．

4.2 各 MODE および MODE 間の処理

プリ・プロセッサーは，DECLARATION MODE 中のプログラムを，ドラム内における，テーブル形式のデータに変換する．DECLARATION MODE が始まると，これらのデータは，いったん，磁気テープにオブジェクトである FORTRAN プログラムによって受理できる形式で，掃き出される．このとき，次のような制限が付け加えられる．

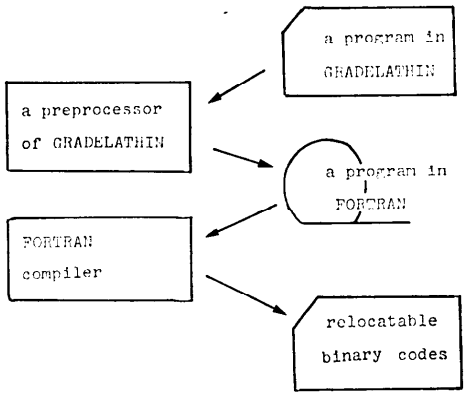


Fig. 4 Processing GRADELATHIN

(1) DECLARATION MODE 中に使用された，アレー・ネームは，唯一つの類に属さねばならない．たとえば，アレー・ネームとして，A (7) を使用したとき，A (1) を SPACE-name に A (2) を，POINT-name にといった具合に，混合することは許されない．混合を許せば，プリ・プロセッサーにおける変数名登録用ルーチンは大変複雑なものになることが予想されるからである．

(2) DECLARATION MODE 中に定義されるアレー・ネーム，および変数名と PROCEDURE MODE 中に定義されるアレー・ネーム，および変数名とは，SPACE-name, 実数名を除いて，完全に独立である．すなわち，POINT-AP などに用いる変数名 AP は，PROCEDURE MODE 中では，全く異った意味を持つ変数名として使用できる．

これらのゆえに，前者の MODE と後者の MODE を何らかの手段を用いて結合する必要がある．この目的のために，プリ・プロセッサーは，オブジェクト・プログラムに自動的に EQUIVALENCE ステートメントを付け加える．たとえば，両方の MODE で，××なる実変数名を使用したとする．このとき，DECLAPATION MODE 中の××は，オブジェクト・プログラム中の実変数 REALN (ⅰ) に変換される．REALN (ⅰ) 表は，プリ・プロセッサーが，オブジェクト・プログラム中に，固有領域として，設定するものである．PROCEDURE MODE の変換の始めに

EQUIVALENCE (REALN (ⅰ), ××)

としておけば，2つのモードの変数××は，同じ意味

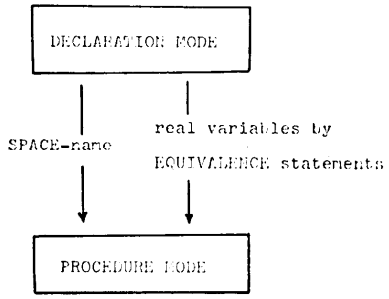


Fig. 5 Information transmission

に使用できる.

なお, SPACE-name は, PROCEDURE MODE 中でも, 使用可能である. ただし, DISPLAY ステートメント中にのみ許される. 各々のグラフィック・ネームは, テーブル形式のデータに変換され, ネームには, そのテーブルの番地が割り付けられる. SPACE-name においても, 例外ではない.

DISPLAY ステートメントを実行するサブルーチンにおいては, SPACE-name が導出するすべてのグラフィック・テーブルが走査される. SPACE-name はこの時の起点を指示するものである. このステートメートの他の引数は, GSP を持つグラフィック記述に関する制限を記述するのに用いられる (File-name, Element-name, File-size など). hidden-line 処理用の引数は, DISPLAY ステートメントを実行するサブルーチンのフローを制御する.

このようにして, 2つの MODE を SPACE-name および実変数 (変数, およびアレー) の受け渡しで, 結合することができた (Fig. 5).

4.3 簡単なプログラム・サンプル

簡単なプログラムを付して, GRADELATHIN の説明の一助としたい. 視点を P(1) として, 頂点 P(2), P(3), P(4) を持つ3角形をディスプレーすることを考える. まず, 使用するアレー・ネームを定義する. ディスプレーの対象である3角形は, EPLANE-EP である. このステートメントが持つ, 3つの1は, 3辺が, もし plottable であれば, 出力するという意味を持つ. EPLANE-PS は, 3角形と同じ記述形式をしているが, DISPLAY ステートメントの実行時には, 視平面と解される. これで, 3次元情報の定義をおえ, PROCEDURE MODE に入る. まず, 使用するアレー・ネームを定義する. そして, GSPで定められた, グラフィック用の初期化を行ない, 各頂

```
DECLARATION MODE
DIMENSION PT(7),A(7),B(7),C(7)
SPACE-SP=SURFACE-S1 VIEW-EPLANE-PS VIEW-POINT-PT(1)
SURFACE-S1=EPLANE-EP
EPLANE-EP=POINT-PT(2) POINT-PT(3) POINT-PT(4) 1 1 1
DO I=1,7
POINT-PT(I)=CODX-A(I) CODY-B(I) CODZ-C(I)
EPLANE-PS=POINT-PT(5) POINT-PT(6) POINT-PT(7) 1 1 1
END
PROCEDURE MODE
DIMENSION A(7),B(7),C(7)
    ↕ initialization in GSP
A(1)=10
B(1)=20          } a viewing point (10,20,30)
C(1)=30

A(I)=FUNCA(I)
B(I)=FUNCB(I)    } coordinates(A(1),B(1),C(1)) are
C(I)=FUNCB(I)      determined

DISPLAY(SPACE-SP, 1 , 0, 300, 1)
STOP
END
PROCEDURE END
```

Fig. 6 An example

点の座標値, A(i), B(i), C(i) を計算する. そして, ディスプレーする. この画面のファイル・サイズは, 300ワードである. hidden-line の表示は dotted-line で行なう. その他 PROCEDURE MODE には GSP で許されるすべての記述が可能である.

5. おわりに

プリ・プロセッサーは, 京都大学大型計算センター FACOM-270-30 FORTRAN with GSP を用いて記述した. プリ・プロセッサー用には, FORTRAN 約3,600 枚, I/O のコントロール用に, アセンブラー FASP で約500枚を要した. プリ・プロセッサーは, コアに入り切らないので, 3つのフェイズに分け, それぞれ, 独立の3つのプログラムで構成してある. ドラムは, 32 kW がグラフィック・データ構成用 (グラフィック・テーブル) に使用し, 残り, 32kW は, プリ・プロセッサーの作業領域である. プログラムのコア占有は, 約 22 kW (×3) である. このため, 操作に手間取る. 一部未完成のところもあるので, 実用には耐えないが, すべてを, アセンブラーで書き直せば, 実用性が増すであろう.

なお, 現在のわれわれの図形記述言語では, 回転などの図形操作に関する SURFACE ステートメント中

に，回帰的な表現，たとえば，

SURFACE-A＝ROTARY-RT SUPFACE-B

において，SURFACE-B が SURFACE-A を含んでいたりすると意味を失う．(図形処理ができない．) さらに SURFACE-B の図形操作後を SURFACE-A, SURFACE-C として，これら2つの面A，Cを含む面Dを考えたとき，このDを操作すれば，SURFACE-B を2回操作することになるが，現在では，このような使用法は，禁止される．その理由は，グラフィック情報が，グラフィック・テーブルの形で格納されているので，そのテーブル・スキャンニングには，十分の注意を払う必要があるからである．すなわち，スキャンニング方法に改良の余地があるからである．しかし，これは，なかなか，難しい．

われわれの図形記述言語の DECLARATION MODE では，APT[8] における APT definition ステートメントの一般形式に，かなり近い形式を採用している．すなわち，APT では，

Symbol＝Surface type/Description data

となっているが，われわれの言語では，

Surface type−Symbol＝Description data

となっている点である．しかし，PROCEDURE MODE 中の記述は，APT において許される cutter 動作の記述とは，ほど遠く，ほとんど，FORTRAN であるという点は，全く異質である．しかし，われわれの記述言語で，part の記述を行ない，cutter の軌跡を出力することは可能ではないかと考えるのであるが想像の域を出ていない．

この他にも，A. Hurwitz[9] などの開発した GRAF がある．これら，一つの display variable が，直接ディスプレーを行うレベルの命令の集合を値とすることを基本にして，FORTRAN の中に矛盾なく組み入れたものである．この言語は，図形の記述よりもむしろ，線画のディスプレーを GSP 用いてよりも，より簡単に表記しようとしたものである．すなわち，そこでの display variable は File-name, Element-name に相当する．このことから考えると，われわれの PROCEDURE MODE 中に，この種の拡張を行える可能性がある．この意味での検討は重要であろうと考えられるが，ここでは行なっていない．

また，Kulsrud[10] の GPGL もある．これにおいては，APTにおける図形記述と異って，LINE, POINT などは Surface type ではなく，コマンドになっていて，コントロールがこのステートメントの上にくると，すぐに実行されるという完全なコマンド・タイプのグラフィック言語である．われわれの言語では，図形記述の部分が，完全に分離しているため，その部分に，コントロールがわたることはないので，GPGLとは，全く異質の言語であると言ってよい．しかし，PROCEDURE MODE 中の GSP をコマンド・タイプに置き換えることにより，GPGL の方向への拡張は，可能であると考えられる．GPGL でのコマンドが，ひとつの display variable (GSP での File-name, Element-name) に相当しているので，実現の可能性はある．

DECLARATION MODE にも拡張可能性がある．基本面は，3角形，すなわち，EPLANE であるが，PATCH や QDSURFACE と同程度の複雑さを持つ任意の曲面のステートメントを付け加えることは，可能である．しかし，このときは，この曲面用のテーブルを，一つ増加させ，かつ，DISPLAY ステートメントの実行処理ルーチンに，この曲面用のプログラムを付け加える必要がある．

われわれは，このようにして図形の記述と，その図形の操作可能性に重点をおいて，図形記述言語を作った．まだ，実用には程遠いとしても，図形記述言語のこれからの研究の一助にはなるものと信ずる．

なお Filename, element-nameの使用法は「GSP」[12] に依った．

おわりに，FACOM 270-30 の使用に際し，京都大学計算センターの西原清一氏，天盛晶明氏から種々の援助を得ることができました．ここに感謝の意を表します．課題番号は 5001 YR 060 である．

付録 1 PATCH ステートメントについて

patch 曲面は Coons[7] によれば，曲線座標表示の曲面である．その表現法は，導き方に若干の相異はあるが，Coons, および穂坂[12] により与えられている．たとえば，穂坂の式を用いると patch 曲面 $\boldsymbol{S}(u, v)$ $(0\leq u, v\leq 1)$ は bending 関数 $f(t)$ を用いて，

$$\boldsymbol{S}(u,v)=\boldsymbol{R}_1(u)+\{\boldsymbol{R}_3(u)-\boldsymbol{R}_1(u)\}f(u)+\boldsymbol{R}_2(v)+\{\boldsymbol{R}_4(v)-\boldsymbol{R}_2(v)\}f(v)+\frac{1}{2}f(u)f(v)\{\boldsymbol{R}_{11}+\boldsymbol{R}_{21}-\boldsymbol{R}_{31}-\boldsymbol{R}_{41}\}+\boldsymbol{S}_0(u,v)$$

と表わされる．ここで $\boldsymbol{R}_i(t)$ は i 番目の端の曲線であり，$\boldsymbol{R}_{i1}$ は端点である．bending 関数 $f(t)$ として，何を使うかは，その時々の条件（どの程度，滑らかに隣接する曲面とつながっているか）によって異る．しかし，$\boldsymbol{S}(u,v)$ は，u,v に関する多項式であると考

えても，さしつかえはない．われわれの図形記述に従がえば，境界上の曲線は PROCEDURE MODE 中で，PARAMETER を計算することで与えてやらねばならない．もし，PATCHステートメントをより完全にするのならば，さらに，CURVEステートメントを用意して，

PATCH-(・)=CURVE-(・) CURVE-(・)
CURVE-(・) CURVE-(・) (*) (*) (*) (*)
CURVE-(・)=PARA-(・) @−ε

とすべきであろう．しかし，本論文では，PATCHステートメントを内に持つことが可能であることを示すにとどめてある．このようにして，決定される patch 曲面に対し，hidden-line 処理，輪郭線処理，相貫線処理を行なうのは難かしい．しかし，(u, v) を適当に分割して，$\boldsymbol{S}(u, v)$ を3角形で近似すれば，$\boldsymbol{S}(u, v)$ を3角形の集合とみなすことができるので，それらは可能となる．

付録 2 曲面の近似処理についてのコメント

われわれの用いた2次曲面，patch 曲面は hidden-line 処理，相貫線処理，輪郭線処理が，かなり，面倒である．ところが，この2つの曲面は，2つのパラメータを用いたパラメトリック表示の曲面であるので，システム・サブルーチンで，自動的に，近似3角形に分解することは，比較的容易である．従って，ディスプレーする際の処理としては，3角形近似のサブルーチンを用意すればよく，特別な収束型の処理ルーチンを用意する必要はなくなる．つまり，PROCEDURE MODE 中では，2次曲面，patch 曲面は，ディスプレー用の処理を行なう時以外は，DECLARATION MODE における形式をそのまま保つが，ディスプレー用の処理を行なうときのみ，3角形の集合で近似して，種々の図形処理演算を行なって，ディスプレーする．この意味では，3角形は，基本面であり，2次曲面，patch 曲面は，基本面からなる部分曲面であるということができる．

参考文献

1) P. P. Loutrel: A solution to the hidden line problem for computer-drawn polyhedra, IEEE trans. C-19, No. 3, March (1970), 205–213.
2) G. Galimberti and et al.: An algorithm for hidden line elimination, CACM, Vol. 12, No. 4, April (1969), 206–211.
3) R. A. Weiss; BE VISION, Vol. 12, No. 5, May (1966), 22–27.
4) 上内秀隆：3次元 Hidden-Line Problem について，情報処理，Vol. 11, No. 3, March (1970), 144–154.
5) 上内秀隆：図形処理における輪郭線について，情報処理，Vol. 11, No. 5, May (1970), 274–285.
6) 上内秀隆：3次元 Hidden-Line Problem について (II)，情報処理，Vol. 11, No. 9, Sep. (1970), 504–508.
7) S. A. Coons: Surfaces for computer-aided design of space forms, Project MAC, MAC-TR-41, June 1967.
8) IIT Res. Ins.; APT part programming, McGraw Hill (1967).
9) A. Hurwitz and et al.; GRAF, Proc. SJCC (1967), 533–557.
10) H. E. Kulsurd; A general purpose graphic language, CACM, Vol. 11, No. 4, Arpil (1968), 247–254.
11) FACOM 270-30 GSP 仕様書.
12) 穂坂衛：曲線曲面の合成および平滑化理論，情報処理，Vol. 10, No. 3, May (1966) pp. 121～131.

(昭和46年12月13日 受 付)
(昭和47年11月6日 再受付)