GIL による多面体処理の記述

穂坂 衛 (東大宇宙航空研)　木村文彦 (電総研)
柿下尚武

1. 序

GILを用いて問題を解いた1つの例として、インタラクティブな多面体の構成法について述べる。ここでインタラクティブな多面体の構成法とは、人が計算機を用いて自分の考えを確かめながら逐時、継続的に多面体を構成してゆく事を意味する。仕事の性質上当然の事ながら、最終的に出来上った多面体は、仕事を始める時には予測出来なかった部分を必ず含んでいるが、多面体という性質を持っている事は最初から変りはない。

こういった仕事は計算機利用の型でいうと、内部モデル処理型に属する。従って、先ず解決されねばならないのは、一般の多面体を計算機内部でどのように記述しモデルとするかという問題であり、更に人が多面体構成に用いる手段である組立て、分離、相貫といった操作を、この内部モデルに対するプロシジュアとして確立する事が必要である。次に考えねばならないのは、それ等種々の操作を、人が計算機に指示する手段を確立する事と、指示の結果を人が多面体として様々な情報を確認し易い形式で人に提示する事である。従来問題にされて来た図形入力という問題も、実は、この図形構成の段階で捉えられるべきものである。

以上のような、インタラクティブな多面体の構成を行うシステムを記述する言語に対しては、上に述べた仕事の性質上、次の事が要求される。先ず、多面体の内部モデルが記述し易い事。即ち、アレイ形式のデータをリスト構造で構成するといった事の記述が簡明である事。又、モデルに働きかけるプロシジュアの記述については、アレイ演算が簡明に記述出来ることが要求される。又、処理の途中で新しくnameを作るという事は必ず発生するのであるから、そういったnameの管理を行ってくれると非常に都合が良い。更に、人と計算機のインタフェイスの部分を記述するためには、出来るだけ多種類の入出力動作が記述出来なければならない。

GILという言語は、これら諸条件を満たすべく計画され、発展して来たが、ここでは具体的に入力から出力迄を一貫したシステムとして記述してその評価も試みる。

本論文では、先ずモデルの基本的なデータ構造と、幾何学的関係を利用したいくつかの処理技術について述べる。次に物体の干渉を解く為のアルゴリズムと、出力の隠線消去について述べる。更に、図形構成の入力法の一例として平面の単純化された入力法と、それの内部モデルへの組込み等を例として、GILの記述性を示す。GILを用いた出力図の例も示されている。

2. 多面体の内部記述

先ず説明を簡明にする為の多面体についての言葉の定義と、多面体の性質を与える。多面体とは、平面により規定される表面で囲まれた物体であり、その法線ベクトルはその部分の表面のそれと一致している。ある点と平面との間隔は、その点が平面の法線ベクトル側にある場合を正とする。説明を簡略にするために

もし間隔が正であれば、その平面はその点から'見える'という。これは点と平面との位置関係を示す言葉であり、又者の間に障害物があって実際には見えなくてもそう言う。二つの平面の交線は、又平面の表面の間で測った平面角の優劣によって稜又は谷線と呼ぶ。ある表面の周囲が全て稜か全て谷か、又はその両方で出来ているかによって、その表面はそれぞれ凸か、凹か又は混合面と言われる。同じ種類の面が連なって領域を形作る。凸又は凹領域は常に混合領域により囲まれている。混合領域との境界は、定義から稜又は谷のみから出来た単純ループとなる。表面が全て凸又は凹である多面体は、凸又は凹多面体である。その形状は表面を規定している平面群のデータが与えられれば決定できる。一般的に、凸又は凹領域は必ずしも凸又は凹多面体の一部であるとは限らないが、若し領域の境界線ループに沿って並んでいる面が確定すればその領域の形状も決定出来る。従って、多面体の基本データは、表面を規定している平面の名前、その表面が属している領域の名前、平面を規定する点と法線ベクトル、及び混合面を囲んでいる面の順序付けされたリストである。このリストには、囲んでいる面との交線が稜か谷かを示す情報も含まれている必要がある。他の補助的なデータは、必要に応じてこの基本データから導ける。例えば混合面に対するものと同形式の図1の様な隣接リストを全ての面に対して作るとか、領域接続表であるとか、頂点の名前やデータの表であるとかがその例である。こういったデータ全体が、多面体の内部記述を構成する。インターラクティブな過程では、人は操作を加える度に新しく発生する全てのデータに対して、予じめ名前を与えておく事など出来ない。しかし、名前がついていないデータに対してはアクセスは出来ないから、次の操作をそのデータに加える事も不可能となってしまう。我々のシステムでは、頂点や交線に対して、自動的に内部名(f_i, f_j, f_k)や(f_i, f_j)を与えるようにしている。但し、f_iは関係する面の名前である。こうする事によって、外部名をexplicitに与えなくてもプロシージャはデータアクセスすることが出来る。

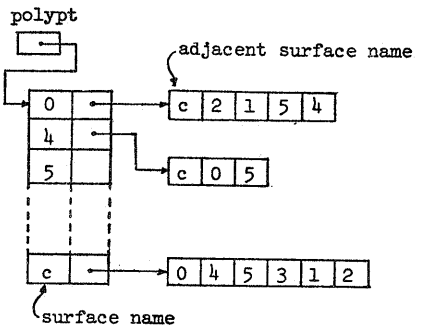


図1. 隣接リスト

任意の多面体は仮想面を用いて相隣る凸多面体に分割出来る。この分割は、谷線を含む仮想面で次々に多面体を切ってゆくことで実現出来る。切断の回数を減らす為に、若し混合領域のうちに面ループを持つものがあれば、他の方法が考えられる。先ず混合領域をその中の稜又は谷のループに沿って二つに分割し、その各々を隣の凸又は凹の領域に接続する。そうすると多面体は概ね凸の領域と概ね凹の領域にわかれる。概ね凹の領域は法線ベクトルを逆にすれば概ね凸の領域に変換出来るので、以下概ね凸の領域について述べる。概ね凸の領域に上述の分割法を適用すると、凸多面体要素に分割出来る。元の多面体をわけたループの一部を含む凸多面体要素があれば、更にそのループのセグメントを含む平面で切断し、その要素が真に凸であるようにする。

こうして、多面体は凸の要素に分解される。切断の過程が進行するにつれ、元の多面体の隣接リストは分離、変更される。この切断の過程は次節で述べる交

線ループ検出の過程の特殊な場合である。この内部記述に対してはどのようなプロシージュアでも作用させる事が出来、その度に記述は新しくなる。こうして、次々に操作を加えてゆけるので、最初の多面体の記述さえも、より単純な多面体を加工して作り出すことが出来る。

3. 幾何学的関係と交線ループの検出

多面体処理においては、二つの多面体の干渉を決定するプロシージュアが本質的に重要である。一般の多面体は凸多面体の結合、削除によって構成出来るので、基本プロシージュアとしては二つの凸多面体の干渉検出が出来ればよい。それについて以下に述べる。

先ず、幾何学的関係を用いた有用なテクニックについて述べる。平面は、単位法線ベクトル$\underline{n}$と原点との間隔eによって定義される。その平面と空間の任意の点$\underline{Q}$との間隔dは

$$d = e + \underline{Q} \cdot \underline{n} \qquad (1)$$

により与えられる。但し、・は内積をあらわす。
以下の表記法で、サブスクリプト付の名前は、そのものが、同じサブスクリプトを持つ面に属している事を示す。図2に示すように、面f_1とf_2の交線(f_1, f_2)に平行なベクトル$\underline{m}$は

$$\underline{m} = \underline{n}_1 \times \underline{n}_2 \qquad (2)$$

で定義される。但し、×は外積をあらわす。交線(f_1, f_2)上の点$\underline{P}$がmだけ動いた時、$\underline{P}$と面f_iとの間隔d_iはw_iだけ変化する。

$$w_i = \underline{m} \cdot \underline{n}_i = \underline{n}_1 \times \underline{n}_2 \cdot \underline{n}_i \equiv [\underline{n}_1, \underline{n}_2, \underline{n}_i] \qquad (3)$$

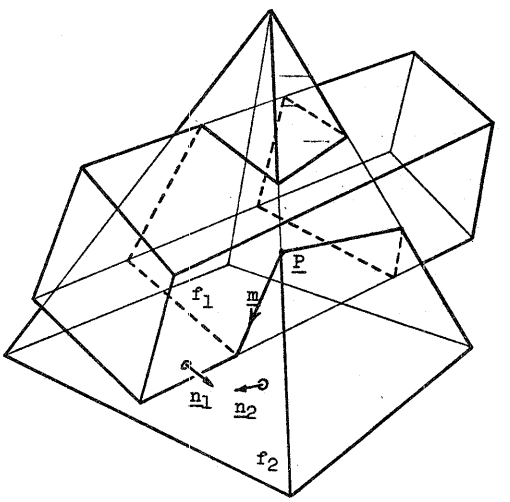


図2. 交線ループの検出

若し、ベクトル$\underline{m}$が面f_i上から始まっているとすると、w_iの符号は$\underline{n}_i$が面f_iのどちら側にあるかを示す。$\underline{m}$が凸の角錐の頂点から発している場合、角錐の側面f_iに対するw_iの符号が全て負であれば、$\underline{m}$は角錐の内部にある。このことを利用して頂点を共有する二つの角錐の干渉を求める事が出来、このテストを角錐干渉テストという。

直線(f_1, f_2)上の一点$\underline{P}$と各面f_iとの間隔をd_iとすると、3面の交点(f_1, f_2, f_i)への$\underline{P}$からの相対ベクトル$\Delta\underline{V}_i$は、

$$\Delta \underline{V}_i = r_i \cdot \underline{m} \tag{4}$$

で与えられる。但し、

$$r_i \equiv -\frac{d_i}{w_i} = \frac{d_i}{\left[\underline{n}_1, \underline{n}_2, \underline{n}_i\right]} \tag{5}$$

である。この点の$\underline{P}$と他の面f_jとの間隔d_jは、交点(f_1, f_2, f_i)に於て

$$\Delta d_j = r_i \cdot \left[\underline{n}_1, \underline{n}_2, \underline{n}_j\right] \tag{6}$$

だけ増加する。面f_iと直線(f_1, f_2)との交点は、(f_1, f_2)上にr_iの値の大きさの順に並ぶ。r_iが負であると、交点は$\underline{P}$に対して$\underline{m}$と反対の方向にあることになる。

二つの凸体AとBとの交点のうち1点$\underline{P}$が与えられると、その点$\underline{P}$から始めて交線ループを決定出来る。その点$\underline{P}$が交線(f_1, f_2)上にあるとする。但しf_1はAに、f_2はBに属するとする。面f_1とf_2を囲んでいるまわりの面は隣接リストを参照すれば判るが、それ等が交線ループと次に交わる面の候補となる。即ち、候補面のうち与えられた点$\underline{P}$からのr_iの値の絶対値が最小である面f_kが次に交わる面であり、点$\underline{P}$と各面との間隔d_iは、点(f_1, f_2, f_k)との間のものに修正される。次の交線は、f_kがBに属するかAに属するかにより、(f_1, f_k)か(f_2, f_k)かになる。r_iのうち同じ最小値のものが2つ以上あった場合、交線はどちらかの物体の頂点を通るか、又は二つの物体の共通の頂点を通る。その場合は、次の交線は候補面間の交線のうち、その頂点を通り、物体表面内にあるものを選ぶ。この過程を繰り返すことにより、交線ループが得られる。この方法は一般の多面体に応用出来、相貫する物体が凸であれば、より単純になる。

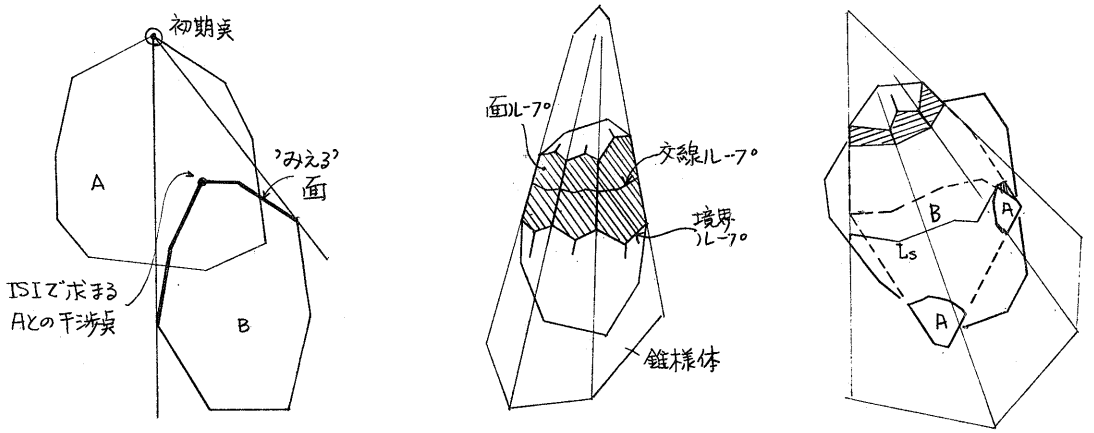
凸多面体の性質としては他にも次の様なものが利用できる。凸多面体を規定している平面のうち任意のものから作り上げた多面体はやはり凸であり、しかも元の多面体を中に含む。交線ループを含む面のループが存在する。この面ループに隣接して次々に面ループが考えられ、最終的には、単純接続された領域で終る。面ループの平面を延長することにより、錐様体が構成出来、この錐様体は元の多面体を中に含む。

ある点$\underline{P}$と、'見える'面との間隔の和はその点と、物体との分離の度合を示す。この値は、点が物体の表面上又は内側にある時に0となる。物体の外にある2点PとQを結ぶ線分がその物体を貫通しない条件は、PとQ双方から'見える'面のうちr_iの値が最大のものと、直線PQとの交点から'見える'面が少なくとも一つあることである。

4. 多面体の干渉

二つの凸多面体の交線を全て求める過程は三つの部分からなる。即ち initial search of interference (ISI) と intersection loop detection (ILD) と final search of interference (FSI) である。ISIでは、二つの物体が干渉す

るか否かを先ず調べ、もししていれば、交点もしくは他の内部にある頂点を一つ求める。交点が一つ求まると、交線ループを求める為にＩＬＤが働く。その過程は前節で述べた。ＦＳＩはまだ干渉する部分が残っているか否かを調べ、もし残っていれば交点を一つ求め、それによりＩＬＤが再び働く。ＩＬＤとＦＳＩは、全ての干渉ループが求まる迄繰り返し用いられる。図3(a),(b)にＩＳＩとＦＳＩを図解してある。



(a) ISI：可視性テスト　　(b) FSI：ループと錐様体の定義

図3. 干渉検出

(i) Initial Search of Interference (ＩＳＩ)

物体Ａの任意の１頂点をとり、可視性テストによって物体Ｂと干渉しているかどうかを調べる。もしその初期点が物体Ｂの外にあれば、その点から'見える'面のうち最も遠くにある面の頂点の一つを選んでそれを新しく初期点と考え、ＡとＢとを入れ替えて同じ過程を繰返す。もし干渉が見つからなければ、両物体上の互いに他の外にあって互いが'見える'一組の頂点が求まる。次にそのうちの一方から始めて、他の物体の各面との間隔の和が最小となる様な頂点を探す。もしその最小値が０であれば、干渉が存在する。もしある頂点で零でない最小値を持ったとすると、その頂点から出ている稜が他の物体を貫くか否かを見る。それでもまだ干渉が見つからぬ場合は、組の他の頂点に対して同じ過程を繰返す。そしてもし干渉がなければ二つの物体は完全に分離されている。もし頂点が他の物体の表面上にある時は、角錐干渉テストを行ない二つの物体が互いに他の外側にあるか、一方が完全に他の内側にあるか、又は互いに相貫しているかを調べる。

(ii) Final Search of Inteference (ＦＳＩ)

交線ループ又は一方の物体の内側にある１点が見つかったら、ＦＳＩが働いて残りの干渉を求める。交線ループL_0を含む面ループの全ての面から錐様体が作られ、内側の物体はそれに包まれる。

錐様体の一つの稜又はその延長が外側の物体と点Ｐで交わるとする。そのＰ

から、錐様体の面が一つでも'見え'れば、外と内の物体にはそれ以上干渉は存在しない。もしPから錐様体の全ての面が'見えなけ'れば、Pから始まる交線ループL_sがILDにより見つかる。交線ループL_oを含む面ループとその隣りの面領域との境界もループとなり、それをl_oとする。もしl_oが完全に外側の物体の外にあれば、L_sが最後の交線ループである。もしl_oとL_sがQで交わるならば、ILDが働き、Qから始まる外側と内側の物体の真の交線ループL_tを見つける。次にL_tをL_oとみてFSIが再び働く。l_oとL_sが交わらない場合には、l_oをL_o, l_1(次の隣接する面領域との境界ループ)をl_oとみて、FSIが再び働く。FSIがリカーシブに働くと、全ての干渉ループが求まる。

ISIの段階で頂点が物体内部にある事が分った場合、その頂点のまわりの面を延長する事で、常に外側の物体と干渉する様な錐様体が出来、ループL_sがそこに作られる。

一般の多面体は、仮想面によって分離された凸多面体よりなるので、上に述べた手順はその各々に対して適用出来る。交線が領域の境界線や切断により生じた仮想面と交わる場合、その交線を規定している面は、隣りの凸多面体にまで延びている。

5. 多面体処理の出力

多面体の内部記述が計算機内に保持されているので、その隣接リストや頂点、平面の値は指定した形式で出力できる。出力として多面体の形を描く場合多面体の内部記述に基づいて隠線消去のプロセスが働く。

隠線消去の問題は、共通頂点を持つ複数個の無限に延びた錐体とその物体の、共通頂点から'見える'面との干渉の問題に他ならない。共通頂点は視点であり、錐体の側面は物体の'見える'領域の境界となる稜又は谷線を含んでいる。与えられた多面体は凸に分割できるので、その境界線ループも全て凸である。ループを見つけるには、隣接表をたどることにより求まる。即ち、'見える'領域の一番外側の面ループの面は必ず'見える'面と'見えない'面の両方をその隣接面として持っているので、隣接リストを見て、それを順次たどってゆけば良い。3節で述べた角錐干渉テストを適用して、交わっている錐体と、他を含む錐体が決定出来る。錐体の側面と、物体の'見える'面との交線はILDを用いて決定出来る。

面の隠れた領域は隠される面領域が他の錐体に含まれるという事と、その面領域のある面が、同じ視線上の隠している面の頂点から'見える'という事から決定出来る。隠されている'見える'領域と'見えない'面と、仮想面とは、出力図には描かない。凹多面体を反転して出来た凸多面体の場合には、他の通常の物体の'見える'面によって隠されていない'見えない'面の部分は描かれねばならない。図4に隠線消去済みの透視図の例を示す。

多面体と点光源が与えられた場合、陰影づけの問題が起る。影の境界を決定する方法は透視図変換の際の隠れた部分の境界を求める方法と同じである。陰影づけされた面と、地上の影が求められたなら、その隠線消去済の面が描かれる。図4に出力例を示す。

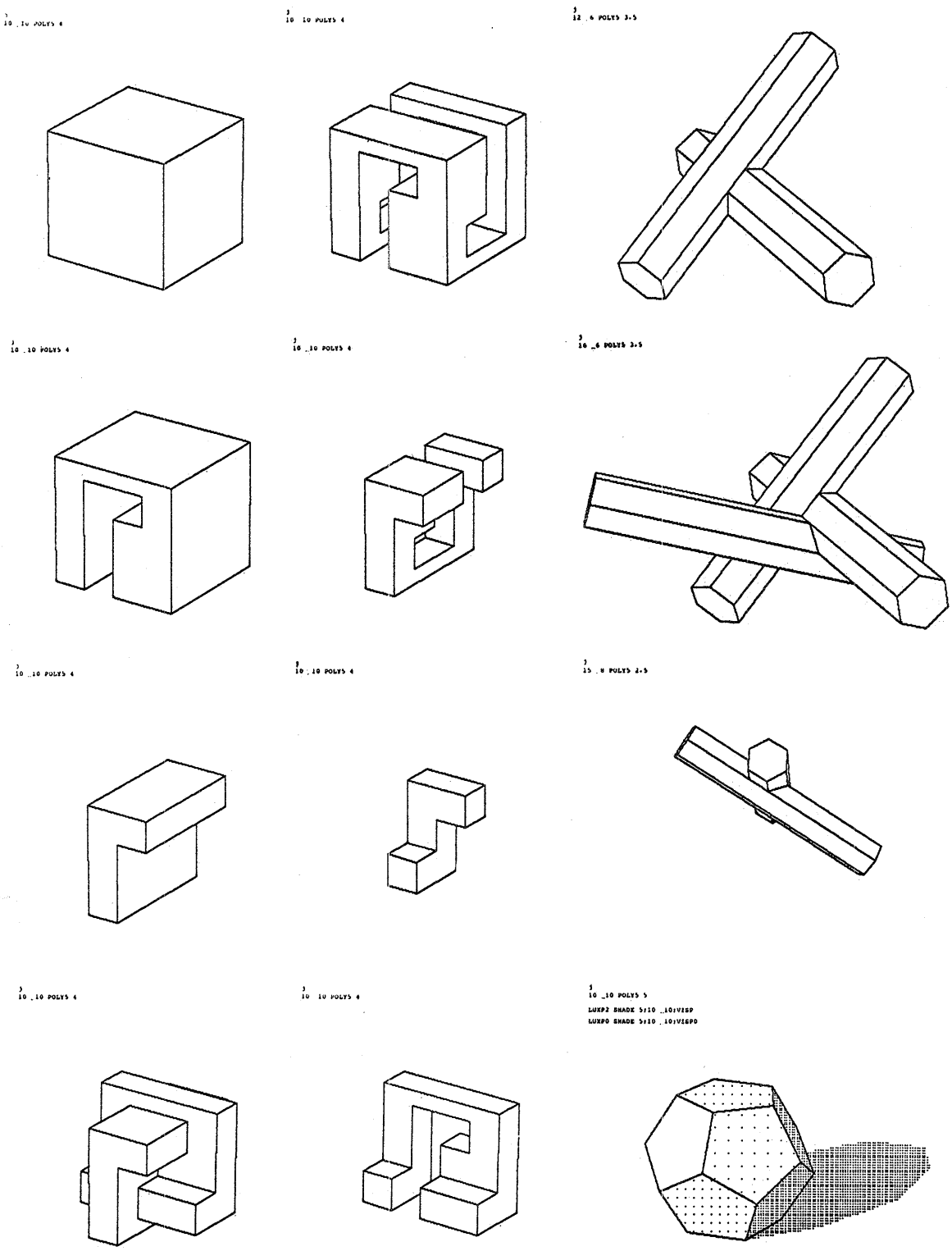


図 4. 立方体のくり抜き，分解，相貫，6 角柱の相貫，12面体の陰影

6. 多面体の入力

これ迄の節で述べた事から，多面体の内部記述が完成すれば，それに対して組立て，分離，相貫という処理を施し，人に見やすい隠線消去済みの図として出力出来る事が示された。一貫した，インタラクティブな多面体構成システムを作るには，あとは如何にして多面体の内部記述を完成するかという問題と，構成の手順をどの様に指示するかという問題を解決すれば良い。

先ず，多面体の内部記述を完成するには，非常に単純な多面体に対して，組立て，分離，相貫の操作を実行すれば良く，問題は，最初の単純な多面体モデルを計算機内部につくり上げる手順を明確にする事に帰着する。結局，初期多面体の内部記述は，多面体の名前と，null のポインターで良く，外部的にはそれが空間全てを埋めている物体であると考えれば良い。それに対して，平面による切断の処理を繰返して，先ず凸多面体を構成し，その組立て，分離，相貫によって，より複雑なものとしてゆけば良い。処理はインタラクティブに行なわれ，多面体を構成してゆく事が即ちそれを入力している事になる。

この事から，入力の問題は構成手順の指示の仕方及び平面のデータをどの様な形式で入力するかという事になり，入出力ハードウェアによって様々な形式が考え得る。しかし，計算機に最も近い所で考えると，信号は常に一次元のストリーム情報になっていなければならず，その意味で，キー・ボードからの入力指示が基本として重要である。

7. GIL による多面体処理の記述

ここでは，キー・ボードから入力した平面を内部表現に変換する処理と，多面体と平面との干渉を解く問題について，GIL でそのプロシジアを記述した例を説明する。例では，説明のため，多少の簡略化が行なわれている。

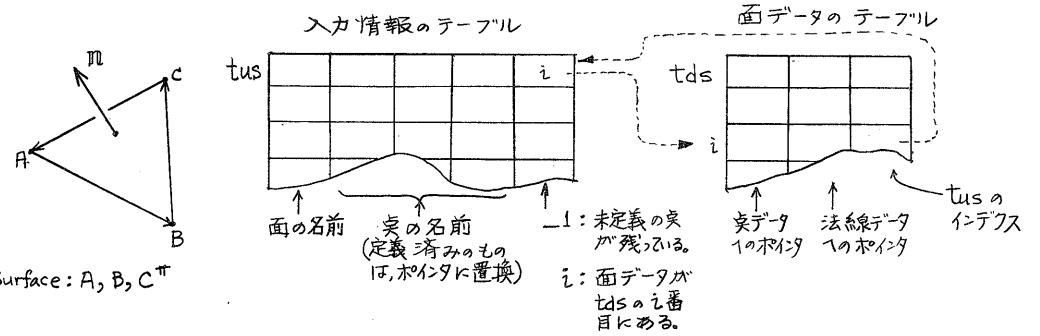


図5. 入力ストリングから内部表現への変換

7.1. 平面の入力

平面は多面体処理システム内では，一点と法線ベクトルで定義されているがこれを直接入力するのは人にとっては容易でない。そこで，キイ・ボードからの入力として考え得る場合最も単純と考えられる方法，即ち3点とその与える順序で面とその向きを定義する方法を例として，外部からの入力を内部表現に変換する手順について述べる。

入力される3点は名前で指定され，その名前に対して座標値が決定した時点

で，法線ベクトルを算出して，面データを作り出す。　この時，システムは2つのテーブル，tus と tds，の更新を行なう。

以上の処理をGILで記述した例について説明する。（図6参照）　マクロ Updttus は，先ず tus を参照し，未定義となっていた点の値が決定しているかを調べ，決定していれば名前のストリングをその値のポインタに置き換える。マクロ文の [1]～[5] がこれに当り，その中に現われるマクロ ⍺ Find β はベクトル β の中のスカラー ⍺ に等しい要素のインデクスを並べてベクトルにしたものを値とするものであり，又 ⍺ Rplctus β は，tus の中の β を全て ⍺ で置き換えるマクロである。　[6] は tus の更新があったか否かを判定して，処理を終了するか，tds の更新を行うかの分岐を行うステートメントである。[7]～[9] で各面の三点が全て定義済みのものを探し出し，[10] で tds に登録済みであるかどうかを調べて，未登録であれば [11]，[12] で登録の手続きをする。　[11] の中のマクロ ✡ はベクトル積を計算するためのものである。又 Length ⍺ は，⍺ の長さを計算するマクロである。

マクロ Input0 は，平面の入力ストリングを解釈して tus を作る。[1] の ⍞ 1 でストリング入力待ちになり，ストリング入力が終了すると，そのストリングが a に割当てられる。　⍺ SrchS β;γ;i はストリング ⍺ の i 番目から始めて，ストリング β と γ で挟まれた部分を抜き出すマクロである。その実行結果は長さ2のベクトルで，第0要素は γ が ⍺ の何番目の要素かを示し，第1要素は，抜き出されたストリングである。

図6.　入力処理マクロの例

```
      @Updttus:$]i;j;n;ix;a;N
[1]   j←0
[2]   →(0=n←⍴ix←_2 Find+a←,tus)/6;i←0
[3]   →(0≠+/0,2=+⍤1 N←a[ix[i]])/5
[4]   →2;⍴(≠⍤1 N)Rplctus N;j←1
[5]   →(n≥i←i+1)/3
[6]   →(j=0)/0
[7]   n←⍴a←+/3=+tus;i←0
[8]   →(3=a[i])/10
[9]   →8*n>i←i+1
[10]  →(_1≠tus[i;4])/9
[11]  tds←((tdsi←tdsi+1),3)⍴tds,(tus[i;1]),(¬N←Length N←({tus[i;
      2]}-{tus[i;1]})✡{tus[i;3]}-{tus[i;1]}),i
[12]  →9;tus[i;4]←tdsi-1
      @

      @Input0:$]v;w;a
[1]   a←⍞1
[2]   v←(w←a SrchS "$";":";0)[1]
[3]   v←v,(w←a SrchS ":";",";w[0])[1]
[4]   v←v,(w←a SrchS ",";",";w[0])[1]
[5]   v←v,((w←a SrchS ",";",";w[0])[1]),_1
[6]   tus←((1+(⍴tus)[0]),5)⍴tus,v
      @
```

7.2　平面と凸多面体との交線ループ

平面と凸多面体との交線ループを求める基本的な考え方は 3. で述べたが，それを実際に GIL で記述するとどの様になるかを，簡単化した例について以下に

述べる。　全体のプロセスは2つに分れ，先ず平面と多面体のある一つの交点を求め，次にその交点から始めて，交線ループを求める。　以下 図7,8を参照しながら説明を進める。

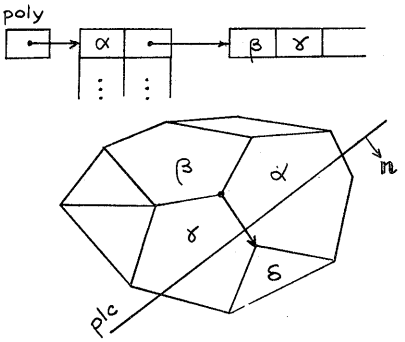


図7.
平面と凸多面体の交点を1つ求める

```
@ Pnt0 : rslt ←⍎plc]adf ; x ; xname;
  e ; e0 ; z
[1]   adf ← {⍎poly}[0 ; 1]} ; rslt ← 0
[2]   x ← Pval xname ← (poly[0 ; 0]), adf[0 1]
[3]   →9 if 0 = e ← *x Dist plc
[4]   →12 if z = xname ← xname Nextv e ; plc
[5]   x ← Pval xname
[6]   →9 if 0 = e0 ← *x Dist plc
[7]   →4 if 0 < e * e0
[8]   →0 ; ptv ← Pval ptn ← plc, xname[1 2]
[9]   →11 if 0 = xname Crstest plc
[10]  rslt ← 1
[11]  →0 ; ptv ← x ; ptn ← xname
[12]  rslt ← _1
@
```

先ず，切断面と凸多面体との交点を求めるマクロ Pnt0 について述べる。このマクロは，正常に交点が求まれば値0を，切断面が凸多面体と接していれば1を，干渉がなければ_1を値としてとる。　又大域変数として，凸多面体のデータ即ち隣接表 poly と面データのアレイが与えられているとする。　切断面 plc が引数である。　又結果として，交点の名前を ptn，値を ptv に作る。　両方とも大域変数である。

[1] で結果の状態を示す rslt を0にする。　次に凸多面体 poly の隣接表の最初の面 α を中心にサーチをするので，その周囲の面を adf にセットする。

[2] で，頂点名 xname を α β γ とし，その座標値をマクロ Pval で求める。

[3] で，頂点と plc との間隔を Dist で求め，その符号を e とし，以後の頂点サーチの指標とする。もし間隔が0であれば，頂点は plc 上にあるので，その処理に分岐する。　[4] で poly の隣接表を用いて，Nextv により，面 plc に対して，元の頂点から発して間隔の変化が e と逆の方向で最大となる方向に頂点をサーチし，頂点 x を更新する。　若し，元の頂点からどの方向に行っても間隔が e の方向にしか変化しなければ，poly と plc とは交わらないからその処理に分岐する。　Nextv では，新しい頂点の名前は，元の頂点を決定するのにも用いられる共通の2面をアレイの1 2 番目におく様に定める。　即ち，図6の例では，δ α γ が次の頂点の名前である。　[5] では，新しい頂点の値を求める。　[6] で新しい頂点の plc への距離の符号を e0 とし，[3] と同様のテストをする。　[7] では e と e0 を比較して同符号であれば新しい頂点への移動では plc を貫通しなかったのであるから，あとは [4] に戻って繰返し頂点を辿ってゆく。　e0 が e と異符号であれば，2つの頂点を結ぶ稜（例では α γ）と plc は交点を持つ。　[8] では交点 ptn を plc α γ とし，値を求めて終了する。　[9] 頂点 xname が plc 上にある場合，plc がそこで poly に接しているだけか，切って

いるのかを Crstest で調べ，切っていれば [11] に分岐する。 [10] では，plc と poly とが接している場合なので rslt を 1 にする。 [11] で現頂点を求める点であるとして，終了する。 [12] では plc と poly とが交わっていないので rslt を −1 にして終了する。

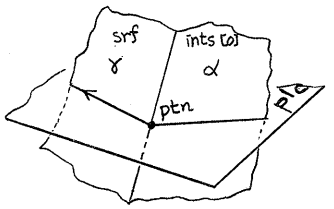
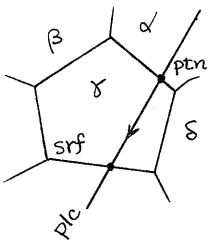
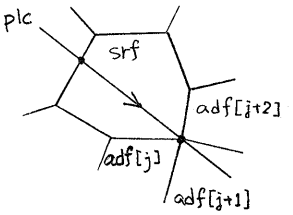


図 8
交線ループを求める

(a) 出発点 ptn



(b) srf から出る点を決める.



(c) plc が頂点を通る場合.

```
@ Loop:$] srf; adf; ds; r; j; jj; i
[1]   ints ← , ptn[1]; srf ← ptn[2]
[2]   ds ← ptv Dist adf ← {{poly}[srf;1]}
[3]   r ← ds Ri plc; srf; adf
[4]   j ← s ⍳ L(0≠s)/s ← |r
[5]   → 8 if s[j+1] = s[j]
[6]   → 0 if ints[0] = adf[j]
[7]   → 2; ints ← ints,srf ← adf[j]; ptv ←
      Pval plc, srf, adf[j]
[8]   p ← plc Intl adf[jj ← j Fan adf]
[9]   → 11 if 0≠+/i ← 0 = ×p
[10]  → 6; j ← p Pyramid srf, adf[jj]
[11]  → 6; j ← (1 Find i)[0]
```

次に，マクロ Pnt0 で得られた ptn, ptv から始めて交線ループを求めるマクロ Loop について説明する。結果として，ints に，ループを形成する面の名前のアレイが作られる。

[1] で ptn[1] (例ではα) をループの先頭にセットする。srf は plc との交線が入ってくる面で，その出口を探すのが問題である。 [2] で adf に srf の隣接面をセットし，srf への入口の点とその面全てとの間隔を求める。 [3] で r を求める。 [4] で j は adf の中で，絶対値最小の r_j を与える面のインデクスである。 [5] 若し，絶対値最小の値を与える面が 2 つ以上あれば plc は頂点を通るので，[8] に分岐する。 [6] 新しい面がループの最初と一致したら終了とする。 [7] で出口の点を求めて次の入口の点とし，ループに新しい面を追加して [2] から繰返す。 [8] plc が頂点を通る場合は，その頂点に集まる面全てと plc との交線を出す。 Fan は，j から始めて，頂点を共有する面全てのインデクスを求める。 [9] 相隣る面との交線が一致したら交線は plc 上にある。 [10] 各交線のうち頂点に集まる面全てからなる錐体の中に含まれるものが新しい面を構成するから，その index を マクロ Pyramid で計算してj として 6 に行く。 [11] 交線を形成している片方の面の adf 内の index を j として 6 に進む。

8 むすび

多面体の処理に関しては，平面を考えの基礎におく事により，内部モデルが簡潔になり，又，処理の方法も，幾何学的関係を用いて整理できた。多面体の処理の問題の基本は，干渉問題であり，これが解けた事により，内部モデルの出力の為の隠線消去の問題も解決出来た。

又，GILの記述性については，アレイ演算が簡潔に書ける事は，この種の問題の記述に向いている事は明らかである。又，リスト・オペレータも，データ構造を取扱う以上，不可欠なものである。名前とストリングの変換も，入力ストリングの形式で新しい名前を定義してゆけるので，インタラクティブなシステムを作るのには都合が良い。