マイクロコンピュータ 24－4
（1982. 11. 2）

# 画像データ操作システム LINKS-DMS

河合利幸 吉村浩 出口弘 西村仁志 河田亨
白川功 大村皓一 （大阪大学工学部）

## [1] まえがき

近年、3次元画像生成の分野における、ハードウェアおよびソフトウェアの進歩により、シェイディングのほどこされた高品位な画像が比較的高速に得られるまでになってきた。これに伴い、問題となるのが、画像データの取り扱いである。シェイディング画像表示は、線画表示に比較してデータの取り扱いという面で次のような解決すべき問題がある。

1) 線画データでは単に点のつながりさえ表現できればよいが、シェイディングのためには、面の表面状態、向き、などの情報が必要であり、データがそれだけ膨大になる。さらに光源という新たなデータも必要である。

2) 高速になったとはいえ、シェイディング画像表示をリアルタイムに行なうことはできず、インタラクティブなデータの定義、特に形状の定義、動作の表現など、が困難である。

LINKS-DMSでは、1)に対して、複雑な形状の物体データを樹木状のデータ構造を用いて表現することにより、視線探索アルゴリズムによる画像生成に適したデータ表現を導入している。一方、2)に対しては、線画表示の高速性を利用して、形状定義の確認のための表示を線画を主体にし、シェイディング表示は必要最小限にとどめる方法がとれるように設計されている。

本文では、シェイディング画像表示による動画作成のためのデータ定義とその修正のためのデータ操作システムの設計について述べる。

## [2] 3次元画像データ構造

視線探索アルゴリズムでは、画素1つ1つに対して多くの計算を行なわねばならないのでその計算の効率を高めるようなデータ構造でなければならない。さらに、動画を作成する時には物体や視点を動かす必要があり、それらの移動が容易に記述できなければならない。前者に対しては、物体をある領域内に箱づめにし、その外側では視線と物体との交差判定を簡略化できるようにしている。後者に対しては、箱づめにされた物体データを木構造にし、物体の動きを親の物体データに対する移動ベクトルと回転マトリクスで表現する。

物体データを、形状リスト（以下Sリスト）、属性リスト（Aリスト）、構造リスト（Cリスト）の3つから構成する。Sリストは物体の形状を記述する式、及び関数からなる。Aリストは色、各反射係数などの物体の表面状態表からなる。Cリストは、Sリスト、Aリスト、及び他のCリストを指すポインタを持ち、さらにCリスト自身が占有する空間情報、及びその正当性を示すフラグ、ポインタがさすリストの座標変換情報、すなわち移動ベクトル、回転マトリクスなどを持つ。これらのリストは図2－1に示すように樹木状に構成される。

### 2－1 形状リスト

Sリストは、現状では3角板、楕円体、濃度分布関数からなる。図2－2にその構成を示す。

3角板には3頂点の法線ベクトルの

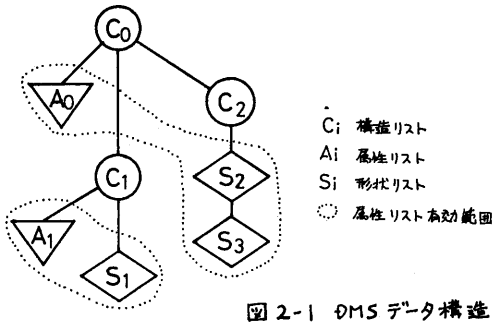



図2-1 DMSデータ構造

```
/****** shape list ******/

/* triangle type1 data structure */

typedef struct
{
int     trtype;          /* 'trtype' defines 'TRI1' */

long    trofs;           /* offset for next shape */

float   ta[ 3 ][ 3 ];    /* position of 3 vertexes */
float   nor[ 3 ];        /* normal vector */
}       TR1DATA,*Tr1data;

/* triangle type2 data structure */

typedef struct
{
int     trtype;          /* 'trtype' defines 'TRI2' */

long    trofs;           /* offset for next shape */

float   ta[ 3 ][ 3 ];    /* position of 3 vertexes */
float   norm[ 3 ][ 3 ]; /* normal vector matrix */
float   t_bu, t_bv;
float   t_du, t_dv;
}       TR2DATA,*Tr2data;

/* ellipse data structure */

typedef struct
{
int     eltype;          /* 'eltype' defines 'ELLIPSE' */

long    elofs;           /* offset for next shape */

float   el[ 3 ][ 3 ];    /* matrix of ellipse data */
float   elrr;            /* elrr = x^2/r1^2 + y^2/r2^2
                                 + z^2/r3^2 */
float   elm[ 3 ];        /* move vector
                            for ellipse center */
float   elr1,elr2,elr3; /* '3 jiku' for ellipse mapping */
float   rtm[ 3 ][ 3 ];   /* matrix for ellipse mapping */
long    texofs;          /* offset for texture mapping */
long    cutofs;          /* offset for cut plane */
}       ELPDATA,*Elpdata;


/* ellipse data option 1 */

typedef struct
{
        long    cutoffs;         /* offset for cut plane */

        float   cuta[ 3 ];       /* cuta[3]={a,b,c} */
        float   cutd;            /* ax + by + cz + d = 0 */
}       CUTPLN, *Cutpln;


/* cloud data structure */

typedef struct
{
        int     cldtype;         /* 'cldtype' defines
                                                 'CLOUD' */
        long    cldofs;          /* offset for next shape */

        float   cg[ 3 ];         /* center of gravity */
        float   rad;             /* radius */
        float   wgt;             /*  weight */
}       CLDDATA,*Clddata;
```

図2-2 形状リスト

```
/****** surface data structure ******/

typedef struct
{
int     sform;           /* surface type */

float   rgb[ 3 ];        /* surface color */

float   dic;             /* diffuse illumination */
float   drc;             /* diffuse reflection */
float   src;             /* specular reflection */
float   sep;             /* shiny exponent */
float   trc;             /* transparent coefficient */
float   rix;             /* refract index */

char    incname[ 20 ];   /* filename of incident
                                    function value */
}       SURFACE,*Surface;


/****** attribute list ******/

typedef struct
{
int     atttype;         /* 'atttype' defines'A LIST' */
SURFACE surface;         /* surface data */

long    mapofs;          /* map list offset */
char    mapname[ 20 ];   /* map list filename */
}       ALIST,*Alist;


/****** mapping data structure ******/

typedef struct
{
int     maptype;         /* 'maptype' defines 'M_LIST' */

char    map_name[ 20 ]; /* map_list filename */
long    map_ofs;         /* offset for mapping */

char    att_name[ 20 ]; /* a_list filename */

int     m_form;          /* mapping type */

float   u_map[ 3 ];      /* u map vector */
float   v_map[ 3 ];      /* v map vector */
float   r_map[ 3 ];      /* mapping origin vector */
int     u_ofs, v_ofs;    /* mapping offset */
int     un, vn;          /* number of texture */
}       TRMAP1,*Trmap1;


/****** incident function file ******/

typedef struct
{
        float   inclim[ LIMNUM ];
        float   incvar[ 2 ][ LIMNUM ];
}       INCFUNC,*Incfunc;
```

図2-3 属性リスト

補間のしかたのちがいにより3種類ある．ひとつは平面分割3角板で3頂点の法線ベクトルはすべて同一で補間の必要がないもの，2つ目は曲面分割3角板で，3角板全領域にわたり補間するもの，もう1つは3角板の稜線にゆみをもたせるため，稜線近傍だけ補間をするものの3つである．後の2つはデータ構造としては同一である．

各形状データに共通して存在するオフセットにより，次の形状データへのポインタを得る．オフセットが0のとき，NILである．

## 2－2　属性リスト

Aリストは表面状態表のほかに、貼り付け用データへのポインタ、入射角を変数とした反射係数関数の関数値表のあるファイル名、物体の材質などが含まれる。図2－3にAリストの構成を示す。

Aリストの有効範囲はAリストAiが直接つながっているCリストCiの下位につながるリスト内であり、下位のCリストCjに別のAリストAjが接続されていれば、Cjの下位に接続されているリストではAjが有効となる。(図2－1)

```
/****** cluster list ******/

typedef struct
{
int     clstype;          /* 'clstype' defines 'C_LIST' */
char    clsname[ 20 ];    /* cluster name */

int     boxflg;           /* box flag */

long    attofs;           /* offset for a_list */
char    attname[ 20 ];    /* a_list file name */

float   bminx;            /* box area */
float   bminy;
float   bminz;
float   bmaxx;
float   bmaxy;
float   bmaxz;

int     amount;           /* amount of data */

long    clsndofs;         /* node offset */
}       CLIST,*Clist;

/****** list node pointer ******/

typedef struct
{
int     nodetype;         /* 'nodetype' defines 'NODE' */

float   crot[ 3 ][ 3 ];   /* cluster rotation matrix */
float   cmov[ 3 ];        /* cluster parallel movement */

long    nxtndofs;         /* offset for next node */

long    ndoffset;         /* node for cluster or shape */
char    ndname[ 20 ];     /* node file name for cluster
                             or shape */
}       LISTNODE,*Listnode;
```

図2－4　構造リスト

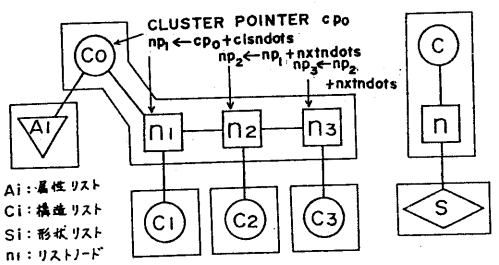



図2－5　リストの接続

## 2－3　構造リスト

図2－4にCリストの構造を示す。親のリストに対する子のリストの座標変換情報をもたせるため、リストノードが設けられている。Cリストを参照する場合のキーは構造名による方法が主体である。この名前はファイル名と共通となっており、物体の名前として用いられる。オフセット及びファイル名によるつながりを図2－5に示す。

例として人の画像データの表現を図2－6に示す。

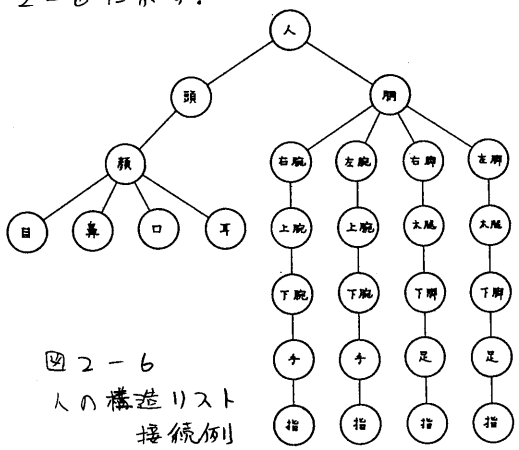



図2－6
人の構造リスト
接続例

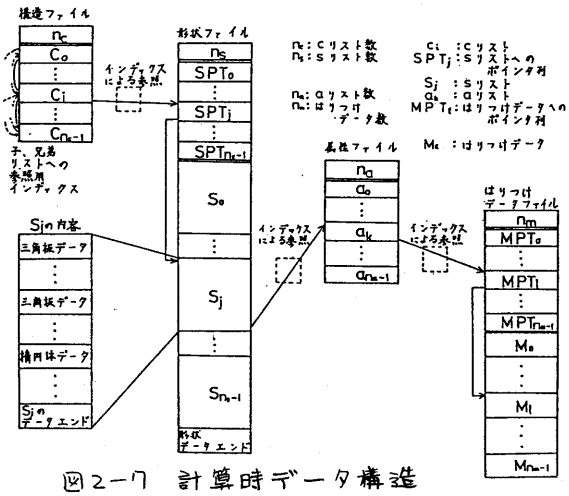



図2－7　計算時データ構造

## 2－4　計算時データ構造

今まで述べてきたデータ構造は、DMS内での表現であり、画像生成の計

算時には効率化のため図2－7に示すデータ構造に変換しておく必要がある。

## [3] 物体形状定義

画像データのうち物体の形状を定義する最も基本的な単位を基本形状という。さらに2つ以上の基本形状を結合することにより、複合物体の形状の定義ができる。ここでは、この物体形状定義の方法について述べる。

### 3－1　3面図入力

3面図を用いた物体形状の入力について述べる。入力装置はディジタイザを用い、ディジタイザ上の3面図の各点を指示することにより図形を入力する。入力結果は3面図としてモニタに表示される。また、必要に応じ3面図のうち1面のみ表示、あるいは拡大、縮小表示等が可能である。しかし3面図では3次元物体の形状を把握するのは難しいので、思いどおりの形状になっているか確認するため、透視図表示が可能となっている。透視図は線画もしくはシェイディング表示の2つが選択可能である。できあがった物体形状は名前をつけたファイルとして登録される。また、登録されたファイルを読み込み、修正も行なうことができる。

#### 3－1－1　入力モード

入力モードは大きく分けて2つある。1つは平面入力モード、もう1つは曲面入力モードである。すなわち3次元物体は面の集まりとして表現される。

1) 平面入力モード

平面入力モードでは、まずどの平面で多角形を入力するのかを指定する。3次元空間での3点を指定することによりこの平面を決定する。さらに、必要な点を入力していき、最初の3点と合わせて多角形を定義する。最初の3点は、もし、全く新しい点であれば3面図のうち2面から指定する。最初の1面で2つの座標成分を決定し、もう1面からの指定で最後の座標成分を決定する。残りの点は平面上にあることがわかっているので1面からだけの指定で3つの座標成分を決定できる。さらに最初の3点も、既に入力された点や辺上、あるいは面上の点

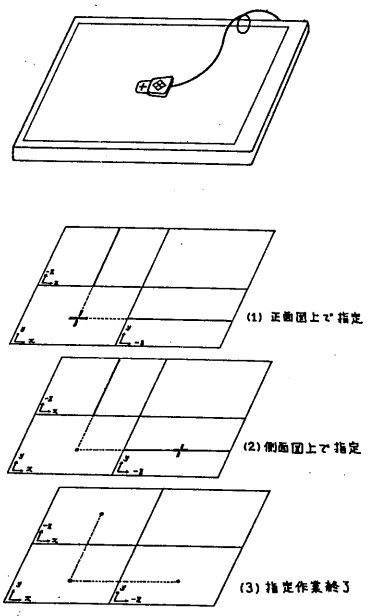


図 3-1. 新しく入力する点の指定

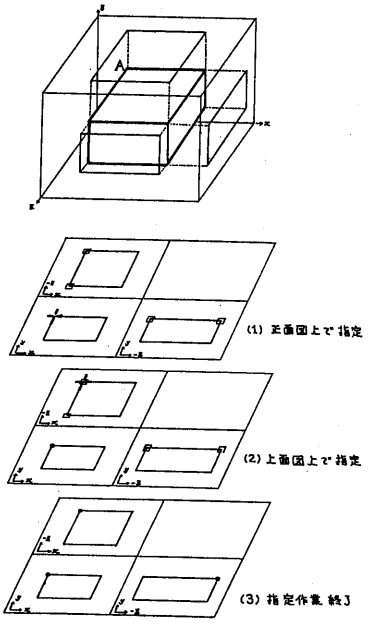


図 3-2. 既に入力されている点の指定

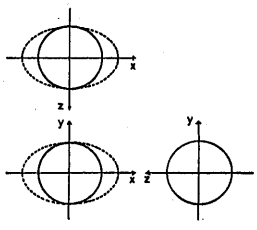


図3-3 楕円体入力

であれば、1面からだけで指定できる場合がある。もし、1面からある点を指定したときに、実際には2点以上が対応するときには残りの2面図上で対応する点が明示され、さらに指示をうながすようになっている。これらの点指定の方法は平面入力モード時だけでなく、点を指定する必要のある時すべてに共通である。

2) 曲面入力モード

曲面はいくつかの3次元パッチの集まりとして表現する。1枚のパッチは次のように定義する。

i)格子状の節点を入力する。曲面はこれらの節点すべてを通る滑らかな面として定義される。

ii)3次スプライン関数を用いて節点間を格子状につなぎ、各節点でのu, v方向の傾きを求める。

iii)格子の4すみの節点の座標と、u, v方向の傾きにより、Fパッチを生成する。これを各格子で行なう。すなわち、1枚のパッチはFパッチの集まりとして定義される。

3-1-2 コマンド

3面図入力には、スケール、原点、精度指定コマンド、点指定コマンド、多角形入力コマンド、消去コマンド、点移動コマンド、曲面入力コマンド、表示指定コマンド、ファイル関係コマンドなどが用意されており、ディジタイザで選択する。

3-2 楕円体入力

楕円体入力モードでは図3-3のような3面図入力と同様な表示がでる。最初は球が、その中心を原点とする座標軸とともに表示される。次に楕円体の各軸の長さを各座標軸上の点をディジタイザで指定することにより決定する。各軸の長さが決まると新しい楕円体の3面図が描かれる。次に必要に応じて回転、切断を行ない希望の形状にする。ただし、各軸の長さは楕円体の各軸と各座標軸が一致していないと変更できない。

切断は、3点を3面図のうち2面から入力することにより指定し、この3点を通る平面で楕円体を切断する。3点を指定すると切り口が表示され、残したい部分の輪かく上の1点を指定すると、他の部分は消え、その部分だけが残る。

さらに希望する時にはシェイディングで表示で形状を確認することができる。

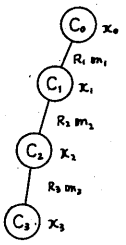
## [4] 接続

物体形状定義で作成した物体はごく基本的なもので、さらに複雑な物体を作りたいときには、これらを互いに連結する必要がある。これが部品接続である。また、一枚の画像を作成するためには、定義した物体を空間に配置する必要がある。これがカット定義である。このカット定義は、動画を作る際のキーフレームを定義するのにも用いられる。

いずれの場合も物体データの操作という面からみれば、Cリストを木構造に連結するという点で共通している。その際に必要な情報は、どの物体をどの座標系に対してどう回転、移動して接続するかということである。

まず物体の連結であるが、新たにCリストを作りその中のリストノードに子の物体リストのファイル名を入れることで実現される。この親のファイル名が連結して作られた物体の名前となる。リストノードは可変であるから、親のリストには子のリストをいくつでもつなぐことができる。

次に物体の接続関係であるが、すでに述べてきたように、Cリストのリストノードには、親のリストの座標系に対する子のリストの回転マトリクスと




Ci : Cリスト
xi : Ciの局所座標系での位置ベクトル
Ri : 回転マトリクス
mi : 移動ベクトル

図4-1 物体の座標変換

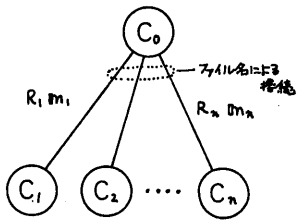



図4-2 Cリストの接続

移動ベクトルが用意されている．図4-1にこの関係を示す．親の座標系への変換は次式で与えられる．

$$x_{i+1} = R_i x_i + m_i$$

すなわち2つ以上の物体を接続するということは，図4-2に示すようにRi miを決定することである．

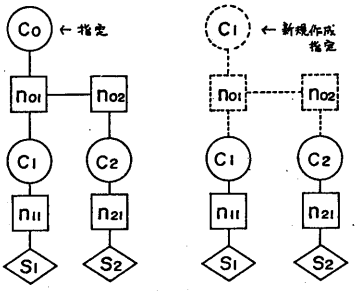



図4-3 接続

## 4-1 接続操作および表示

まず，親となる物体名（Cリスト名）を指定する．このとき既にある物体名を指定したときには，修正モードとなり，その名を持つCリスト以下につながっている物体に対する回転マトリクス，移動ベクトルの修正，すなわち物体の接続関係の修正が行なえる．新しい名前を指定したときには接続モードとなり，つづいて，接続する物体名を指定する（図4-3）．

表示は指定した物体固有の座標系で行なわれるが，下位のレベルの座標系に切り換えることもできる．さらに何レベルまで表示するかも指定できる．表示方法は，線画，物体の占める領域を表わす直方体（box表示），各局所座標系の座標軸方向とその接続関係（skeleton表示），シェイディング表示の4通りが用意されており，目的に応じて選択する．

接続関係の指定は，表示とは独立した座標系で行なえる．指定した物体のリストの親のCリストに対する相対座標が自動的に採用される．このとき，Sリストのつながっている1つのCリストの下位には新たなCリストをつなぐことはできない．指定は，物体を示し，ディジタイザで回転角，移動量を指示し，表示を見ながら行なうことができる．このときにはシェイディング表示はできない．ディジタイザでは

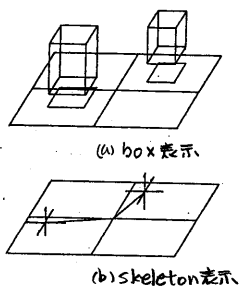



図4-4 表示方法

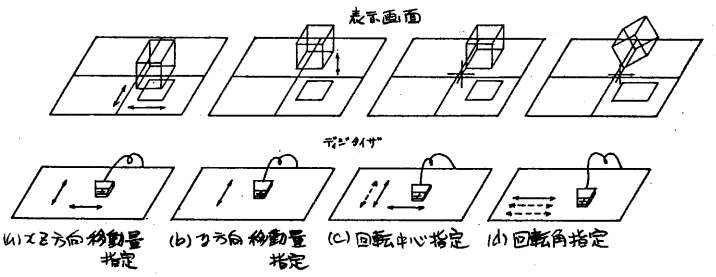



図4-5 接続指示

2次元量までしか指定できないので切り換えて行なう（図4－5）。

4－2　視線定義

作業をしているうちに、見る場所を変えたいという要請がでてくる。このときは、物体の移動と同様に視点（カメラ）も移動できる。このときの視点の空間情報は表示に用いられている座標系に対するものとなる。物体の移動と異なる点は、目標点を指定し移動でき、さらにその目標点を固定したまま視点を移動できることである。目標点を指定せずに場所を移動すると、視線方向が変化せずに視点だけが移動する。指定の方法は図4－5に示す物体の移動と同様に行なう。

また、視点位置はそのままで、カメラの焦点距離を変える（ズーミング）こともできる。ズーミングは広角レンズや望遠レンズを使うのと同様な効果を得ることができる。指定は0～90°までの画面への投影角を示す（図4－6）。さらにカメラを視線方向まわりに回転することもできる。

これらの視点の空間情報はファイルに登録しておくことができ、いつでも呼び出し、使用できる。

画像生成時には、視点が原点となるよう、カット定義されたデータが座標変換され、計算が行なわれる。

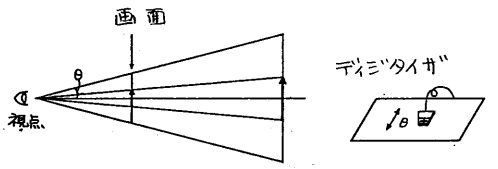



図4－6　ズーミング指定

4－3　光源定義

シェイディングを行なうためには、物体の表面状態のほかに、光源を決定する必要がある。この光源の各データを決定するのが光源定義である。

光源には、平行光線、点光源、スポットライトの3種類があり、種類により指定するデータが異なる。一方、色指定はいずれの場合も共通で、後に述べる表面定義の色指定と同様に行なう。平行光線は、カット中の2点をディジタイザで図4－5（c）と同様に指定し、その方向を定義する。点光源もカット中の2点を指定し、その位置および方向を決める。スポットライトは、光源位置、光を当てたい場所、スポット半径の3つを指定する。

定義した結果は、シェイディング表示で確かめられるが、線画でも、各光源位置、方向などを希望により表示させることができる。

［5］動作定義

太陽と地球と月の運動を考えてみる。地球は太陽のまわりを回転し、月は地球のまわりを回転している。ここで、太陽に対する月の運動を記述することを考えると、文章では表現しにくいことに気づく。文章で表現できないことはすなわち、人間にとって考えにくいということである。ここまで考えてくると、物体の運動を相対的に、しかも階層的に記述することは、人間にとって自然で、表現しやすいということがわかる。すなわち、物体を木構造を用いて表現し、相互の関係を親に対する子の座標変換を用いて表現することは、運動の記述にとって、必然的なものといえる。

動画を作るには、1秒あたり30ないし24フレームの連続した画像を作らねばならないが、すべてのフレームのデータを4章まで述べてきた方法で作るのは無理がある。そこで、従来のアニメーションを作る時の手法にならい、あるシーンのキーとなるようない

くつかのフレーム（キーフレーム）を定義し（キーフレーム定義）、このキーフレーム間をコマ割りして動画を作る。

キーフレーム定義は、4章で述べたカット定義が基になる。接続操作のうち、修正モードを用い、希望する定義されたカットの各物体の接続関係を修正し、物体、視点、光源を動かしたカットを作り登録する。これを順次繰り返し、キーフレームを作っていく。

このようにしてできた各キーフレームの各物体、視線、光源の各データを、指定されたコマ数でスプライン補間しあるシーンの全フレームのデータを生成する。さらに1シーンの最後にはその終端条件が出力される。これは次のシーンへの接続に使われる。

補間するデータにより、部品操作、形状変形、視線補間、光源操作の4つに分けることができる。

## 5－1　部品操作

物体データのうちCリストのリストノードの回転マトリクス、移動ベクトルが補間すべきデータである。Sリストの物体データの相互関係は変化しない。すなわち、物体は変形しない。

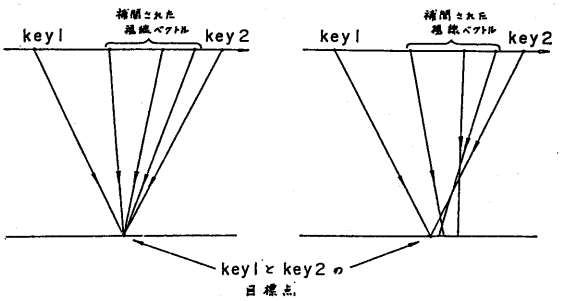
## 5－2　形状変形

物体形状定義で使用した各データを変え、物体を変形させる。たとえば、楕円体であれば、各軸の長さを変化させる。

まず、変形する物体を接続操作の時と同様に指定すると物体形状定義のモードとなり、必要な変更を行なうと、そのデータが補間され、形が変化する。

## 5－3　視線補間

視線の補間方法には2種類ある。1つは目標点の座標補間から視線ベクトルを補間する方法で、常に指定したところが画面中央にくるようにしたい時に使う。もう1つは、目標点の座標は補せずに、視線ベクトルの補間を行なう方法である。これら2つのちがいを図5－1に示す。




(a) 目標点補間　　(b) 視線ベクトル補間

図5－1　視線補間

## 5－4　光源操作

各種光源によって補間方法がちがうが、光源の色の補間は共通である。平行光線ではその方向を、点光源およびスポットライトでは視線と同様な2通りの補間、さらにスポットライトではスポット半径の補間がある。

以上の補間に関して指定を要する項目は、キーフレームの定義とともに、補間データとしてファイルに登録する。さらに動作定義の結果は、線画表示（動画）で確認することができる。[3]

# [6] 表面定義

物体の表面状態の指定は、動画作成の工程では最終的なキめであるが、本システムではいつでも必要なときに指定できる。指定せずにシェイディング表示を指示した時はデフォールト値で表示するようになっている。

ここで定義したデータはAリストとしてファイルに登録される。ある物体にある表面状態を定義することは、物

体に対応するCリストにAリストのファイル名を入れることで実現される．

### 6－1　色入力

色立体中の一断面を表示し，必要な色を選択し，ファイルに保持できる．また，複数色の色対比なども画面上で可能である．色の選択および色立体中の移動の指定などはディジタイザで行なう．

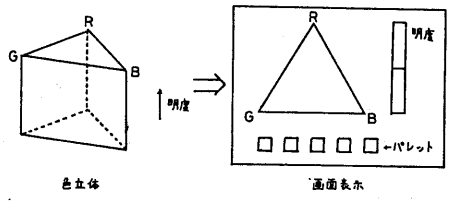



図6－1　色入力

### 6－2　表面係数入力

図2－3のAリスト中の物体の表面状態を表わす6種類の係数，すなわち，周囲光反射係数，散乱反射係数，直接反射係数，直接反射拡散指数，透過率，屈折率の値を指定し，入力する．これらの値は物体の質感を左右する．その確認は円柱もしくは球のシェイディング表示で確認できる．このとき，光源の色指定も可能である．

### 6－3　貼り付けデータの入力

貼り付けデータの入力は，画面をキャンバスとしてディジタイザで描き，入力する．ぬりつぶし，階調をつけた入力なども可能である．

## [7] あとがき

3次元画像データの入力，修正を支援するためのシステムLINKS－DMSの設計について述べた．これまでは，物体の座標データを人手により計算して求め，数値をキーボードから直接入力する方法をとってきたが，物体形状などが思いどおりでなかった場合，再び計算しなおすという，非効率的で繁雑な作業をしいられてきた．LINKS－DMSにより，これらの作業は大幅に軽減され，数値を直接入力することはほとんどなくなった．さらに，入力したデータ，特に形状，動画における動きなど，線画の高速性により容易に確認できるようになり，動画作成の期間短縮などの利点をもたらした．

しかし，入力の手間が軽減されたとはいえ，入力装置がディジタイザとキーボードのみなので，まだまだ入力は繁雑である．入力すべきデータは3次元量であるのに，ディジタイザでは2次元量しか一時に入力できないからである．ディジタイザの補助としてジョイスティックの接続や，3次元量が直接入力可能な装置の開発を通して，入力をより容易に行なえるようにするのが今後の課題である．

[参考文献]

1) David F. Rogers, J. Alan Adams, 山口富士夫訳，「コンピュータグラフィックス」，日刊工業新聞社，1979
2) 山口富士夫，「コンピュータディスプレイによる図形処理工学」，日刊工業新聞社，1981
3) 中山，平井，大野，西村，江木，河田，白川，大村，「画像生成用マルチマイクロコンピュータ」，情報処理学会マイコン研資，1982．11
4) 吉村，辰己，西村，河田，白川，大村，「LINKS－1における画像生成手法」，情報処理学会マイコン研資，1982．11