グラフィクスとCAD 18－3
（1985. 7. 5）

マルチコンピュータシステムMC－1における3Dアニメーション

3D Animation of the Multicomputer System MC-1

平井 誠　日高 教行　浅原 重夫　鷲島 敬之
Makoto Hirai　Noriyuki Hidaka　Shigeo Asahara　Takayuki Sagishima
松下電器産業㈱ 無線研究所
(Matsushita Electric Ind. Co. Ltd., Wireless Research Lab.)

The 3D animation language of the multicomputer system MC-1 is described. The basic objects are triangle, rectangle, ellipsoide and cut-plane, and the basic operations are move, rotation, translation, scaling, coloring, etc.. The objects are constructed in a tree structure. The objects and the operations are treated in variable form, so that action, metamorphosis and change of color can be handled easily.

1．まえがき

コンピュータグラフィクスは広い分野でその有用性が高まりつつあるが、その中にコンピュータシミュレーションの視覚化がある。コンピュータシミュレーションの膨大な出力結果を映像化して表現すれば、かなり効果的にシミュレーション結果を理解できる。特に3次元空間において時間的に変化するデータを3Dアニメーションにより映像化すれば、極めて大きな情報伝達速度が得られる。しかし、複雑な形状をもつ物体の複雑な動きをスムーズに表現するには、多くのコマからなるアニメーションを作る必要があり、コンピュータの演算量は膨大なものになるため、高速の画像生成システムが必要となる。

我々は、高速浮動小数点演算ユニットを付加したユニットコンピュータ（MC）か

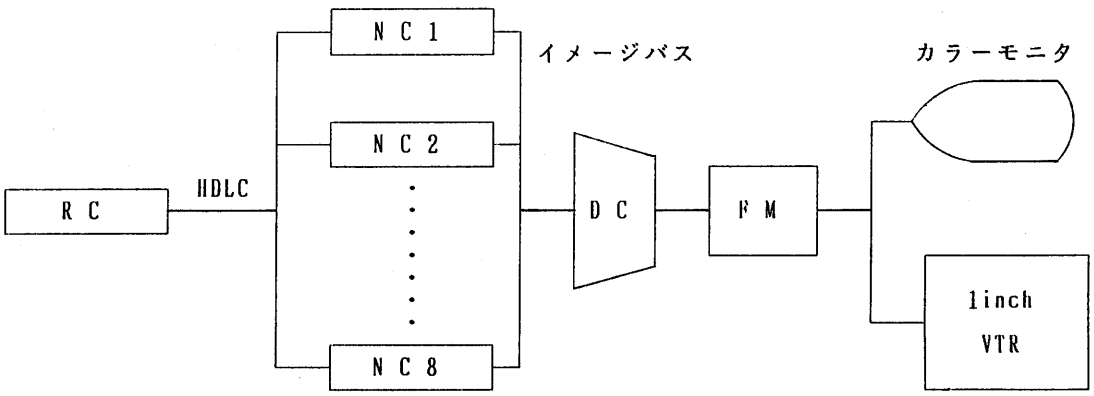


図1　MC－1　システム構成図

ら構成されるマルチコンピュータシステムＭＣ－１を開発し、[cl3 c23]３Ｄアニメーションの研究を現在行なっている。本文では、ＭＣ－１における３Ｄアニメーションについて報告する。

本アニメーションシステムの主な特徴として、以下の点が挙げられる。

① ＭＣ－１において効率よく画像生成できるデータ構造を提供すること。

② ３次元形状，表面属性，形状の構造及びそれらの時間的変化を容易に記述しかつ操作できること。

③ データのライブラリ化が容易なこと。

④ 移植性が良く、拡張性に富むことである。

2．システム構成

図１はＭＣ－１のシステム構成図である。ＭＣ－１はルートコンピュータ（ＲＣ）とノードコンピュータ（ＮＣ）がＨＤＬＣ回線（４Mbps)により星状接続されたマルチコンピュータシステムであり、ＲＣの監視のもとに各ＮＣで共通の画像生成プログラムが実行される。各ＮＣにおいて生成された画像は高速イメージバスを経てフレームメモリ（ＦＭ）に出力され、ＣＲＴに表示され、1inch ＶＴＲによりコマ録りする。

ＲＣにおいては、ユーザーとのインターフェース，アニメーションのデータ作成操作，高速イメージバスとフレームメモリ及びＶＴＲの制御等も行う。

ＮＣは１６ビットマイクロコンピュータ６８０００（ＭＰＵ）と高速浮動小数演算ユニット（ＡＰＵ）から成り、ＡＰＵは浮動小数点乗算器，浮動小数点加算器，関数演算器（ＳＩＮ，ＣＯＳ，逆数，平方根），９６ビット×２Ｋワード×２バンクのＡＰＵプログラムメモリ，２４ビット×４Ｋワード×２バンクのデータメモリを持つ。２バンクメモリ構成により、ＭＰＵとＡＰＵは並列に動作可能であり、また各浮動小数演算器は命令９６ビットのＡＰＵプログラムにより並列動作可能である。ＭＣ－１では、Ｚバッファ法とレイトレーシング法に基づいた画像生成手法を用いているが、処理効率を上げるために、以下の手法が用いられている。

① ３次元空間内で近接する形状データを一まとめにし、それらを内包する直方体のデータ（ＢＯＸ）を付加することにより、画像生成する際に、光線と直方体とが交差する場合のみ、その直方体が内包する形状との交差判定を行なえばよいので、処理量を減少させられる。

② ある直方体Ａとある光源Ｂとの間に位置しその直方体Ａに影を落とす可能性のある直方体のリスト（影リスト）をあらかじめ作成しておき、直方体Ａに光源Ｂの影が落ちるかどうかの判断を行なう際の処理量を減少させられる。

3．アニメーション言語

３次元アニメーションを作成するには、３次元形状，表面属性，形状の構造，視点の位置，およびそれらの時間的変化を容易に記述かつ操作できるアニメーション言語が必要である。以下、ＭＣ－１におけるアニメーション言語について述べる。

3.1　シーン記述

3.1.1　形状記述

現在ＭＣ－１では、三角板、平行四辺形板、パッチ三角板、パッチ平行四辺形板、楕円体、切断面、透明切断面の７つの基本形状を組合わせて利用できる。図２に基本形状の記述形式を示す。パッチ三角板は、３角板の面内での法線ベクトルを、３頂点において与えられた法線ベクトルを線形補間して求めることにより、表面が滑らかに屈曲した陰影表示を可能にする。切断面は楕円体を切断し、切断面を見せるものであり、透明切断面は、切断面を見せず、楕円体の内側を見せるものである。'写真１参照。点光源では、光源位置とその強度を指定する照点位置を入力する。

3.1.2　属性記述

表面属性として、色、反射属性、透過属性に分けて記述する。図３に属性記述形式を示す。色は表示装置の赤緑青の各色の最も明るい色を色ベクトル（１，１，１）で、最も暗い色を（０，０，０）で表わす。光源強度も色ベクトルで与える。反射属性は、周囲光反射係数re，散乱反射係数rd，金属反射係数rm，金属反射指数ri，鏡面反射係数rsで表わす。　透過属性は、透過係数ｔ、屈折率ｎで表わす。

3.1.3　構造記述

構造記述を図４の文法で行う。図５にシーン記述の一例を示す。

#movコマンドは、１つ上の階層に対する相対移動を示す。

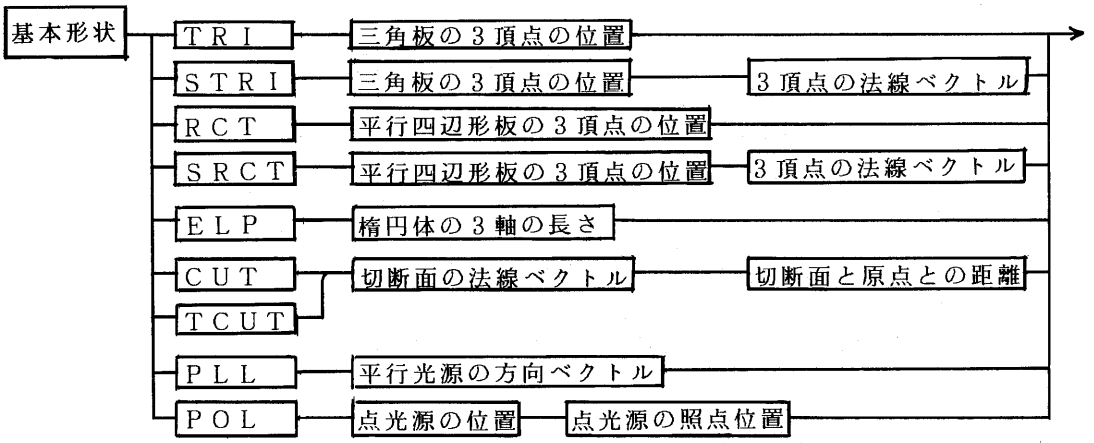


図２　基本形状の記述形式

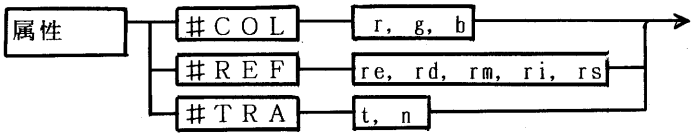


図３　属性の記述形式

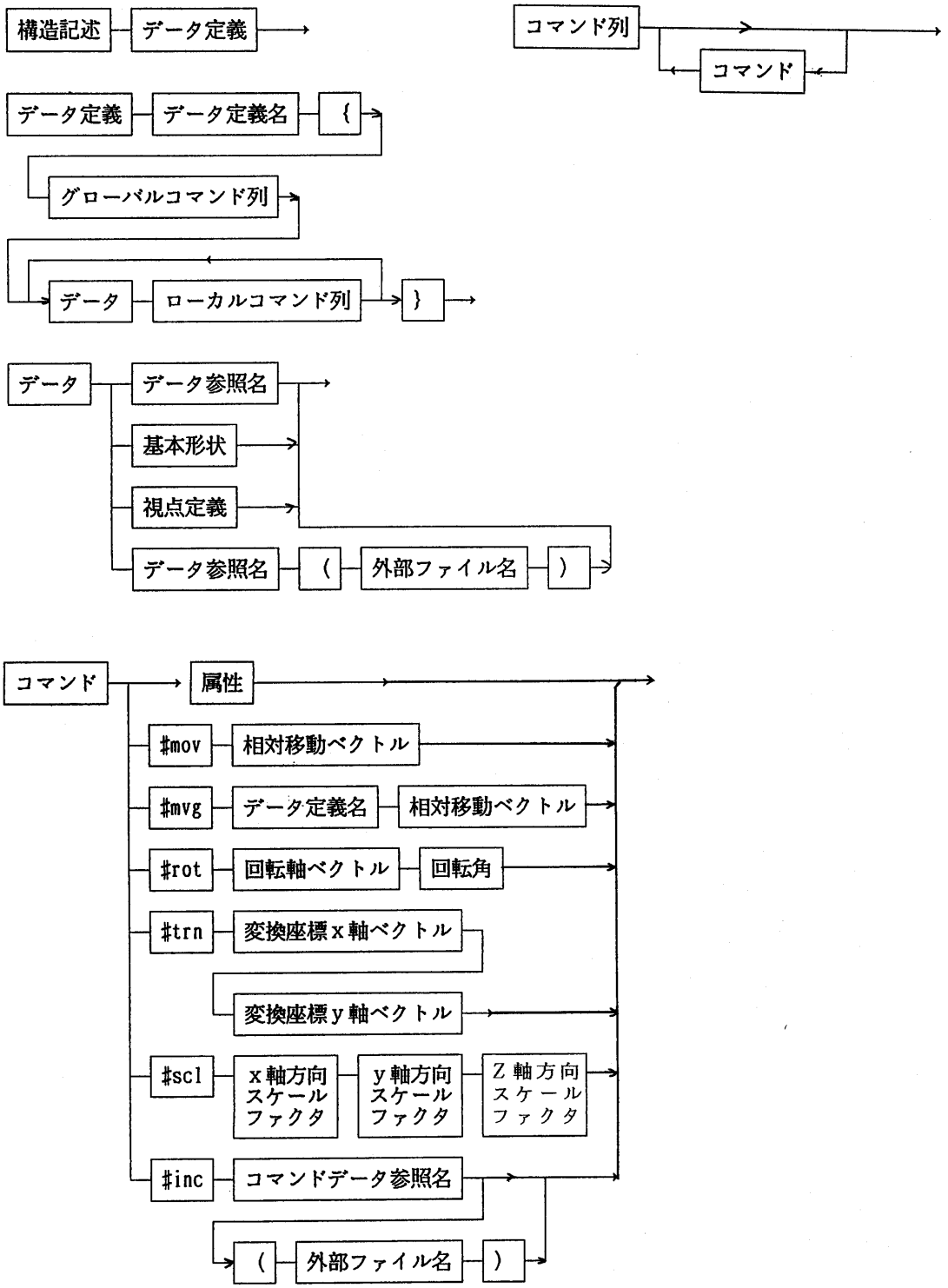

構造記述
データ定義
コマンド列
コマンド
データ定義
データ定義名
{
グローバルコマンド列
データ
ローカルコマンド列
}
データ
データ参照名
基本形状
視点定義
データ参照名
(
外部ファイル名
)
コマンド
属性
#mov
相対移動ベクトル
#mvg
データ定義名
相対移動ベクトル
#rot
回転軸ベクトル
回転角
#trn
変換座標ｘ軸ベクトル
変換座標ｙ軸ベクトル
#scl
ｘ軸方向スケールファクタ
ｙ軸方向スケールファクタ
Ｚ軸方向スケールファクタ
#inc
コマンドデータ参照名
(
外部ファイル名
)

図４　構造の記述形式

#rotコマンドは、１つ上の階層の座標系における回転軸まわりに指定した角度の回転を示す。

#trnコマンドは、１つ上の階層の右手座標系において指定したｘ軸ベクトル，ｙ軸ベクトルをその階層の右手座標系のｘ軸，ｙ軸とする座標変換を示す。

#sclコマンドは、１つ上の階層のｘ軸，ｙ軸，ｚ軸上の単位ベクトルを各軸方向のスケールファクタ倍したものを、その階層の各軸上の単位ベクトルとする座標変換を示す。スケールファクタが負の場合その軸に関する反転を表す。

#mov,#rot,#trn,#scl コマンドは、１つ上の階層に対する座標変換を示すが、#mvg,#rtg,#trg,#scg コマンドは、指定した階層に対する座標変換を示す。

物体を階層構造化して記述することは、以下の利点をもつ。

① 同じ形状を、色を変え、拡大縮小し、反転し、伸縮させて、あるいは元のまま、複数個記述する場合、形状記述は１度でよい。よって物体形状をライブラリ化できる。動作記述もライブラリ化できる。

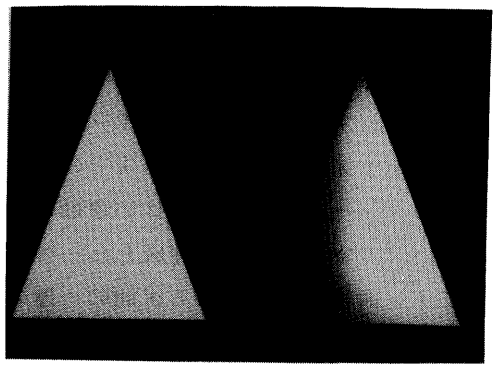
```
tr{
#col 1 1 1
#ref .3 .4 .4 4 0
     pll   -1 -1 -1
     pll   -1 1 -1
     tri
-5 5 -100  -9 -5 -100  -1 -5 -100
     stri
5 5 -100   9  -5 -100   1 -5 -100
 0 1 1      1 -1 1       -1 -1 1
}
```

（ａ）三角板とパッチ三角板

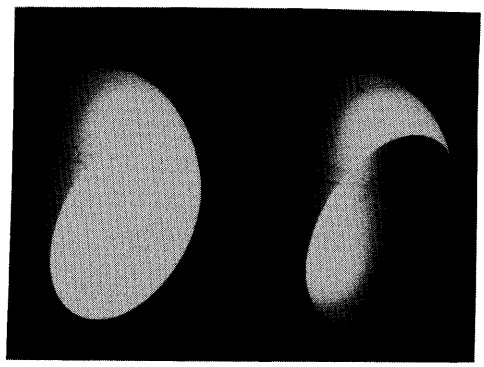
```
el{
#col 1 1 1
#ref .3 .4 .4 4 0
     pll   -1 -1 -1
     pll   -1 1 -1
     el1   #mov -5 0 -100
     el2   #mov 5 0 -100
}
el1{  elp   4 5 4
      cut   1 -.8 1.2 2
}
el2{  elp   4 5 4
      tcut   1 -.8 1.2 2
}
```

（ｂ）切断面と透明切断面

図５　シーン記述例



（ａ）三角板とパッチ三角板



（ｂ）切断面と透明切断面

写真１　基本形状

② 物の動作、変形、変色の記述が容易である。例えば、図6の階層構造において階層c，dが階層bに対し相対位置だけ与えられていると、階層bを動作、変形、変色すれば、階層c，dも同じ動作、変形、変色を行う。階層cにおいて表面属性が与えられている場合、階層cは階層bの変色に従わない。階層cにおいて階層aに対し相対位置が与えられている場合、階層cは階層bの動作に従わない。

③ 物体の構造グラフから、物体間の隣接関係を判断できるため、画像生成時に物体間の位置関係の判断を効率よく行える。

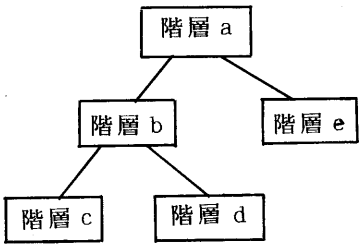


図6　階層構造

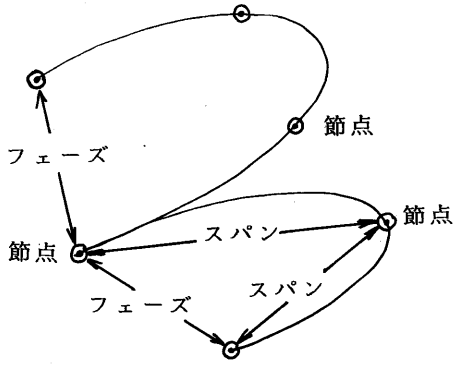


図7　スプライン曲線による運動記述

3.1.4　視点記述

視点は、視点位置、視対象位置、スクリーンの上方向、スクリーンの大きさによって記述される。視点は階層構造化された物体の任意の階層に、複数個設定でき、画像生成時にそれらのうちから適当な視点を選択して画像生成を行う。これにより、ステレオカメラの3D空間内の運動記述、複数の視点からの画像の同時生成による3D物体の表示（例えば3面図、透視図、スタジオのカメラのシミュレーション）、複雑な動作をする物体上からの周囲の光景の生成（例えば、ロボットの手の先についたカメラからの光景）等が容易に実現できる。

3.2　アニメーション記述

アニメーション作成では、キーフレームの記述を行ない、キーフレーム間の中割りにより全フレームの記述を得る。[C3]

3.2.1　定義

フィールドは、1/60秒間に表示される映像であり、フレームは 1/30 秒間に表示される映像である。コマはフィルムに出力される時にフィルム上の1コマに撮影される映像である。VTRでは1フィールドが1コマに当たる。イメージはアニメーション中で静止している一枚の映像である。図7に示すように、スプライン曲線等で物体の運動を記述する場合、スプライン曲線を指定するある節点から次の節点までの区間の一連のイメージを、その物体の運動のスパンという。物体の動作が連続した一つのスプライン曲線で記述される区間の一連のイメージを、その物体のフェーズという。視点の運動が連続している一連のイメージをカットという。同一のシーン記述から作成

された一連のイメージをシーンという。つまり、シーン記述において、基本形状およびコマンドのパラメータに変数が用いられていれば、シーン記述は一イメージのデータではなく、そのシーンの登場人物、舞台装置、大道具、小道具とそれらの位置関係を示しているデータとして扱われる。また同一のストーリー記述から作成された一連のイメージをストーリーという。

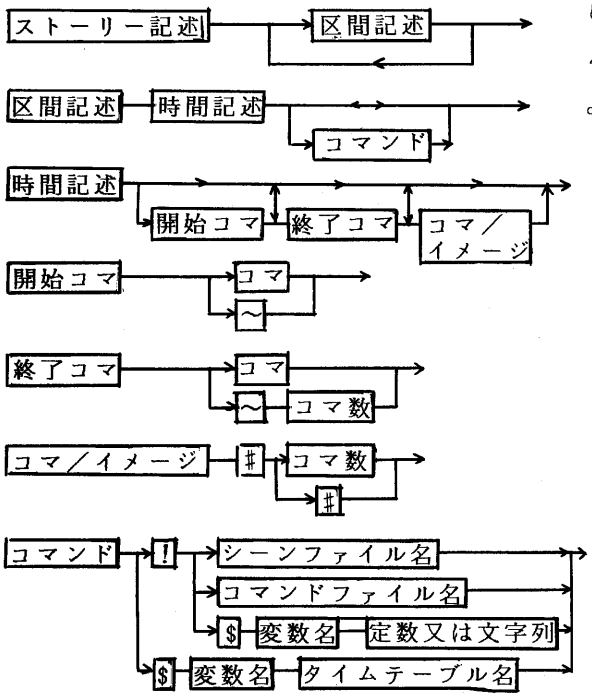


図８　ストーリー記述形式

3.2.2　ストーリー記述

図８にストーリー記述の文法を示し、図９にストーリー記述の一例を示す。図９に対応するストーリーを図１０に示す。

3.2.3.　アニメーションの作成手順

アニメーション作成は、以下の手順で行う。図１１参照。

①　カット及びフェーズの設定。これによりストーリー全体の枠組を決定する。つまり、絵コンテやストーリーボードに相当する。[C3] これに従ってストーリー記述を行う。

```
0   ~1:00   ##   !scene1                  } 区間1
                 $cam   cut1_1
                 $mov   phase1
~   ~2:00   #2                            } 区間2
~   ~2:00   #2   $cam   cut1_2            } 区間3
~   ~2:00   #2   !scene2                  } 区間4
                 $cam   cut2
                 $mov   phase2
~   ~1:00   #4   !scene3
                 $cam   cut3_1
             ##  $mov   phase2
~   ~1:00   #2   $cam   cut3_2
~   ~1:00   ##
```

図９　ストーリー記述例

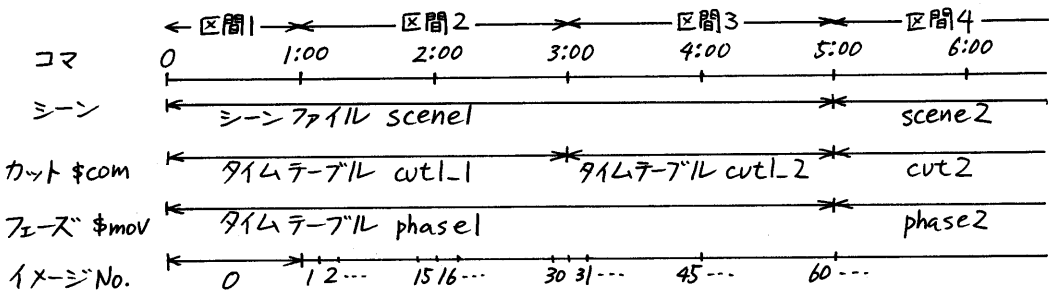


図１０　ストーリー

② 各カット及びフェーズに必要なシーンデータを組み立てる。時間変化の必要なパラメータは変数にしておく。

③ 各カット及びフェーズの変数のタイムテーブルを作成する。キーフレームを定めるタイムテーブルを作成し、線形補間あるいは各種スプライン補間により中割りする方法と、ジョイスティック等を用いて線画表示により、リアルタイムシミュレーションを行う方法がある。

①のストーリー記述と②のシーン記述の作業順序は逆になることも多い。タイムテーブルはシーン記述の構造化によりライブラリ化が容易である。例えば、ボールのはずむ動きとボールの変形、振子の動き、落葉の散る動き等をライブラリ化しておき、適当に時間スケーリング・空間スケーリング・移動・回転することで、各フェーズに合わせられるので、何度もタイムテーブルを作り直す必要はない。

3.3 アニメーション言語処理系

以上のアニメーション言語の処理系の全体の流れを図１２に示す。

アニメーション・ソースファイルは変数を含むシーン記述のファイルで、アニメーション・コマンドファイルは変数を含むコマンドファイルである。このコマンドファイルはＶＴＲの制御プログラム、画像生成プログラム等に対するコマンドを集めたもので、アニメーションの自動生成・自動編集等を行う。

動作軌道ファイル(*.spl) は動作軌道上の節点位置を列記したもので、各種スプライン関数発生プログラムにより、指定された速度で軌道上を動く動作点の各イメージ毎の位置を列記した動作点位置ファイル(*.spp) が出力される。同様にして、速度ベクトルファイル(*.spv) 加速度ベクトルファイル(*.spa) 、軌道接線ベクトルファイル(*.spt) 、軌道主法線ベクトルファイル(*.spn) 等が得られる。これらのファイルから、軌道上を動く物体の位置、姿勢、変形等を制御する。図１３参照。これらのファイルは前述のタイムテーブルとして扱われる。

ストーリーファイルには前述のストーリー記述が入っており、アニメーションジュネレータ (ag) にイメージNo.とストーリーファイルを入力すれば、そのイメージを生成するのに必要なシーンソースファイルとシーンコマンドファイルが生成される。

シーンソースファイルは移植性と拡張性を考慮して文字ファイルとしたが、物体データをライブラリ化するためと、以後の処理を簡単にするために、データコンパイラ (idc)により、バイナリ形成のシーンデータファイル (*.iid)に変換する。この時

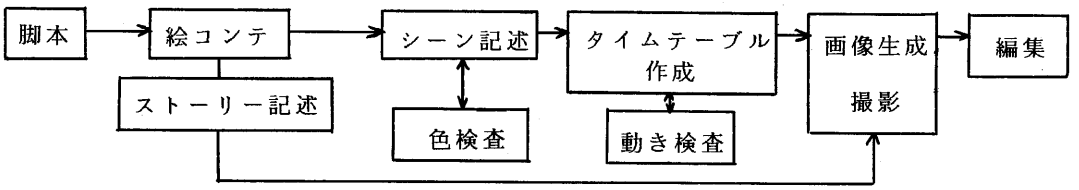


図１１ アニメーションの作成手順

、ソースファイル中のデータ定義にランダムアクセスするためのデータ定義表と、外部のデータライブラリを参照するための外部データ参照表とが付加される。

図１４に、データライブラリの一部とその引用例を示す。データライブラリに定義された立方体（box)をスケーリング（#scl）し、位置決め（#mov）を行っている。

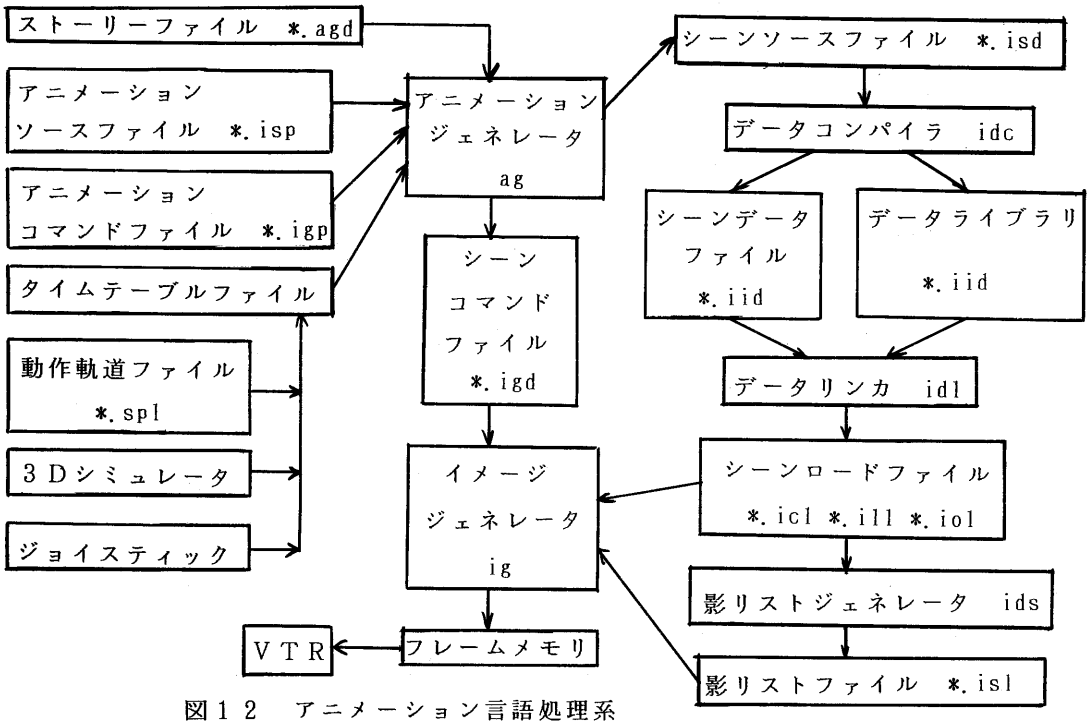


図１２　アニメーション言語処理系

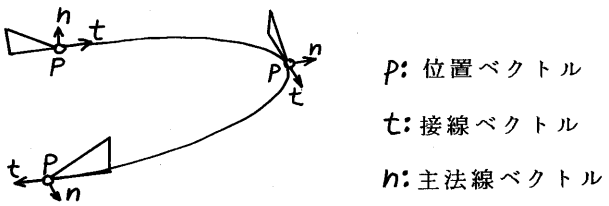
fly.isp アニメーションソースファイル

```
flying{ fly #trn  $t $n
             #mov $p     }

fly{ tri 0 0 0  -3 0 0  -3 1 0 }
```

fly.agd ストーリーファイル

```
$t fly.spt
$n fly.spn
$p fly.spp
```



p: 位置ベクトル

t: 接線ベクトル

n: 主法線ベクトル

図１３　軌道上の物体の姿勢制御

dlib.isd データライブラリの一部

```
box{ plx
     plx #mov 1 0 0
     ply
     ply #mov 0 1 0
     plz
     plz #mov 0 0 1 }

plx{ tri 0 0 0  0 1 0  0 1 1
     tri 0 0 0  0 0 1  0 1 1 }

ply{ plx #rot 0 0 1 -90 }

plz{ plx #rot 0 1 0  90 }
```

rect.isd 引用例

```
rect{ box(dlib) #scl 3 -1 -2
#mov 0 0 -10 }
```

図１４　データーライブラリ

データリンカ（ide)は指定されたシーンデータファイル中の指定されたデータ定義を根(root)とする木構造のシーンデータを、外部データライブラリを参照しつつ収集し、根のデータ定義の相対座標系（ワールド座標系）に座標変換し、#ｂｏｘコマンドを含むデータ定義を根(root)とする木構造のデータごとに一つの物体データとしてまとめ、それらを内包する直方体（ＢＯＸ）を求め、物体データとともに物体ロードファイル（*.iol）に出力する。物体の表面属性データを形状データごとに出力するのは効率が悪いため、同じ属性をもつ同じ物体内の形状データには同じ属性データNo.を与えておき、属性データは必要な属性データのみを出力している。

光源データは全物体に対し、強い影響力を持つため、光源データは常にアクセス可能な状態で保持しておく必要がある。そこで木構造中に点在する光源データを一つにまとめ、光源ロードファイル(*.ill）に出力する。

物体と光源との位置関係をあらかじめ大まかに調べておくと、以後の処理量をかなり減少できる。そこで、ある物体とある光源との間に存在する物体のリストを作成し、影リストファイル(*.isl）に出力する。

木構造中に点在する視点データは木構造の根のデータの相対座標系（ワールド座標系）に変換され、一まとめにして視点ロードファイル(*.icl）に出力する。画像生成時に視点を選択し、その視点座標系に光源及び物体データを変換した後、画像生成を行う。影リストは座標系に依在しないため変換の必要はない。

4.むすび

ＭＣ－１は動き始めたところであり、現在その評価を行っている。数本のアニメーションの作成をスムーズに行うことができた。本アニメーションシステムは以下の特長をもつ。

① ３Ｄシミュレーションの出力結果（タイムテーブル）を扱いやすい。

② 物体の重心の運動と物体の姿勢の変化とを分けて記述できる。

③ 物体の変形，変色、光源の強度変化，変色を容易に扱える。

④ 複数の視点からの映像の生成が容易である。

⑤ 点光源の強度制御が容易である。

⑥ 物体データ及びタイムテーブルをライブラリ化できる。

⑦ ストーリー記述によるアニメーションのバッチ作成が行える。

最後に、本研究の機会を与えていただいた当研究所　中島所長、三木部長に感謝します。

参考文献

1. 中瀬他、”高速浮動小数点演算機能を持つユニットコンピュータＭＣのアーキテクチャ”、情報処理学会計算機アーキテクチャ研究会資料、1985年６月。
2. 日高他、”マルチコンピュータ画像生成システムＭＣ－１”、情報処理学会計算機アーキテクチャ研究会資料、1985年６月。
3. 安居院他、”コンピュータアニメーション”、廣済堂産報出版、1983年７月。