3次元仮想空間を利用したビジュアライゼーション

中 村 庸 郎*

Visualization using 3－D Virtual Space

Tsuneo NAKAMURA

要旨

Virtual Reality Modeling Language (VRML) ブラウザ Live3D が，WWW（World Wide Web）ブラウザである Netscape Navigator の Version 3.0 に標準プラグインとして付属している。したがって，3次元シーンを VRML により記述すれば，現在普及している一般的なマルチメディア・パソコンを使用して簡単にブラウジングすることが可能である。

本論文では，“WWW 上の3次元仮想空間を歩き回る”という VRML 本来の目的から離れ，“手軽に使える仮想的な3次元空間を利用し多種多様なデータを可視化する”という立場から，VRML の持つ有用な機能の利用例を報告する。具体的には，1) ボリュームデータの Octree 表現を用いた Level of Detail (LOD) の構成，2) LOD ノードと平行投影カメラとの組み合わせによる2次元形状あるいは3次元形状のアニメーション，3) 形状ノードの再利用による回転体の簡易モデリング，である。

1．はじめに

従来より，科学，工学，医学等の様々な分野において，データの可視化（Visualization）のために Computer Graphics（CG）が用いられてきている。特に，物体の形状を把握したり，データの特徴量を理解しやすくしたりする必要がある場合には，3次元 CG が適している。また，物体の形状やデータの特徴量が時間的に変化する場合は，3次元 CG を用いたアニメーションが適している。

3次元 CG は，1) 物体や空間をデータ化するためのモデリング，2) モデリングされたデータに基づいて画像を生成するためのレンダリング，という二つの処理によって実現される。したがって，可視化システムの構築のためには，用途に適したモデリングツール，レンダリングツールを用意する必要がある。

一方，ハイパーテキストの記述用言語 HTML (Hypertext Markup Language) を拡張し，WWW（World Wide Web）ブラウザにより Web 上の3次元仮想空間を自由にブラウジングできるようにする試みが始まっている。1994年5月，そのために提唱されたのが，3次元シーンの記述用語 Virtual Reality Modeling Language（VRML）である。その後，1994年11月には VRML1.0ドラフト，1995年5月には VRML1.0[1] が公式に発行されるという急速な仕様決定が行われ，標準化が進んできた。また，VRML1.0の目標は static な3次元シーンの記述であるが，1996年4月に発行された VRML2.0では3次元シーンの中の物体の動き（behavior）を記述することに焦点が当てられている。

また，VRML1.0対応ブラウザ Live3D が，WWW ブラウザである Netscape Navigator の Version 3.0 に早くも標準プラグインとして付属している。したがって，現在普及している一般的なマルチメディア・パソコンに Netscape Navigetor 3.0 をインストールし，VRML で記述された3次元モデルをブラウジングするだけで，レンダリングされた画像を見ることができる。

本論文では，VRML1.0の持つ有用な機能を利用したデータの Visualization 例を報告する。まず，ボクセルに対応させた Cube の集合によりボリュームデータを表示させる際，ボクセルの Octree 表現に基づき，視点からの距離に応じた詳細さで表示させる Level of Detail（LOD）を構成した。次に，LOD ノードと平行投影カメラとを組合せた場合，視点から距離に対応した物体が透視変換によるスケール変化なしで表示される

＊ 助教授 情報工学科

ことに着目し，2次元形状あるいは3次元形状のアニメーションを行ってみた。さらに，基本となる物体のアフィン変換による軌跡をUSE命令を用いて定義すると，比較的複雑な形状のモデリングが容易であることを回転体の生成例で示した。

2．VRMLによる3次元シーン[1)2)3)]

VRMLによる3次元仮想空間を形成する座標系は，図1のように，スクリーンに向かって右方向がx軸の正，上方向がy軸の正，手前に向かう方向がz軸の正，となる右手系である。

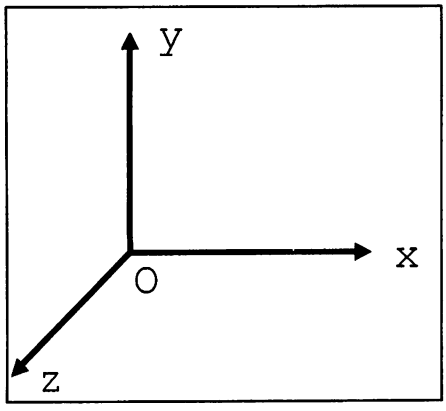


図1　座標系

視点は，仮想的に設置されるカメラの位置に対応し，透視投影（perspective）および平行投影（orthographic）の二種類のカメラが用意されている。両者ともに，デフォルトでは，座標（0，0，1）からz軸の負方向を向くように設置されるが，もちろん変更可能である。

ユーザは，カメラの位置や方向をマウスのドラッグ操作で変化させることにより，あたかも3次元シーンを歩き回り，物体の形状を観察しているような臨場感を得ることができる。「前進」はマウスの左ボタンの上方向ドラッグで行い，z座標が減少する。「後退」は同ボタンの下方向ドラッグで行い，z座標が増加する。また，物体の回転はマウスの右ボタンドラッグにより簡単に行える。例えば，右方向へドラッグすると物体が右方向に回転，すなわちカメラの位置が左方向に回転する。

VRMLによる3次元シーンは，一般に様々なノード（node）のリストで構成される。したがって，シーンの中の物体も一個以上のノードから構成される。ノードには球，円錐，立方体，多角形，点など基本的な形状を記述する形状ノード，前述のカメラの種類・パラメータを記述するカメラノード，照明の種類・パラメータを記述する照明ノード，等さまざまなものがある。

透視投影（perspective）カメラを使用した場合，物体に向かって前進すると物体が徐々に大きく表示され，逆に後退すると徐々に小さく表示されていく。このとき，近くて大きく表示する場合に比べ，遠くて小さく表示する場合には詳細なモデルを使用してもレンダリング処理に時間がかかるだけで無意味である。このように，視点から距離に応じて表示すべき物体のモデルを変更できる機能がLOD（Level of Detail）である。

一方，平行投影（Orthographic）カメラを使用した場合，前進・後退による物体のスケール変化は生じないが，視点からの距離に応じて表示すべき物体のモデルを変更できるLODはやはり有効である。

VRMLでは，リスト1に示すLODノードにより，Level of Detailを実現する。

リスト1　LODノードの書式

```
LOD {
  range [r1, r2, ..., rn]
  center x0 y0 z0

  Separator {
   物体w0の定義
  }

  Separator {
   物体w1の定義
  }

       ...

  Separator {
   物体wnの定義
  }
}
```

この例において，centerフィールドで指定される

$$P_0\ (x0, y0, z0)$$

は，カメラからの距離を計算するための基準点である。いま，カメラの位置P_eを

$$P_e\ (xe, ye, ze)$$

とすると，表1のように表示される物体が選択される。一般に，rangeフィールドにn個の数値が指定された場合，(n+1)個の物体を定義する必要がある。

表1 LODノードにおける表示物体の選択

条件	表示される物体
$\overline{P_eP_0} < r_1$	W0
$r_1 \leq \overline{P_eP_0} < r_2$	W1
...	...
$r_n \leq \overline{P_eP_0}$	Wn

ただし，カメラの初期位置 P_{e0}

$$P_{e0}\ (xe0, ye0, ze0)$$

は，リスト2に示すPerspectiveCamera（透視投影カメラ）ノードのpositionフィールドに指定される。

リスト2 PerspectiveCameraノードの書式

```
PerspectiveCamera {
  position xe0, ye0, ze0
}
```

このLODの仕組みは，OrthographicCamera（平行投影カメラ）ノードの場合も同様である。

3. LODを用いたボリュームデータの可視化

LODノードにおいて視点からの距離に応じて表示すべき物体の定義例として，ボクセルのOctree表現によるボリュームデータへの適用を試みた。

可視化対象となるボリュームデータは，32^3個のボクセルから構成され，各ボクセル V_i は256階調の濃度値 D_i を持つ，さらに，このボリュームデータをOctree（8分木）表現により階層的に記述しておく。ただし，ここでは D_i が256階調であるため，レベルkにおいて濃度値が閾値以下であるボクセルのみを含む場合のみ，レベルk+1への分割を中止することとする。

ボクセル V_i は，VRMLのCubeノードを用いて一個の立方体として表現される。このとき，V_i の濃度値 D_i は，Cubeノードに適用される材質を示すMaterialノード内のtransparency（透明度）フィールドに反映され，その結果として D_i の低いボクセルほど透明に，逆に D_i の高いボクセルほど不透明に描画されることになる。実際には，

$$tr = 1 - \frac{D_i}{255} \tag{1}$$

で求まる値trをリスト3のtransparencyフィールドに指定する。trは0～1の値をとり，0が完全な不透明，1が完全な透明である。

リスト3 Materialノードの書式

```
Material {
  transparency  tr
}
```

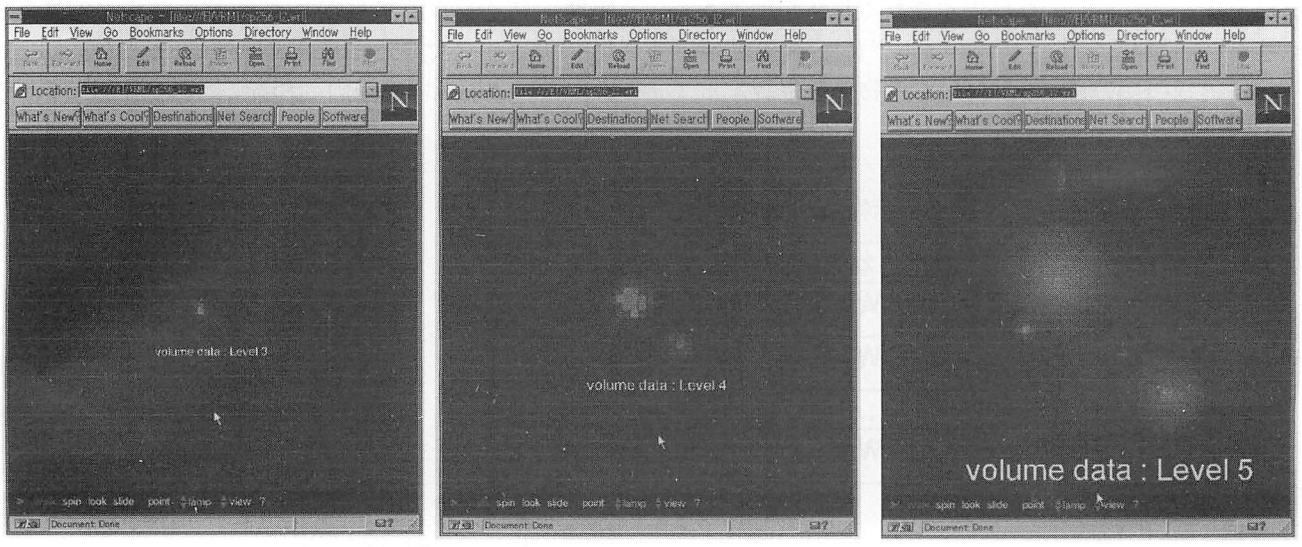
さて，カメラからの距離にしたがい表示の粗さを段階的に変化させるLODのため，Octreeの各レベルに対応した粗さの物体を生成する。つまり，レベルk（k=0, 1, . . . , 5）では$(2^k)^3$個のボクセル空間と考え，各ボクセルに対応する濃度値をOctreeの探索により計算する。このとき，そのボクセルに対するOctreeのノードが存在しないときは濃度値が閾値以下であるから描画せず，存在する場合にはその内部に含まれる全ボクセルの平均濃度値をそのボクセルの濃度値とし(1)式により求まる透明度trをもつ立方体を描画する。

実験の結果，図2に示すように，カメラから遠いほど，小さくかつ薄れて見えるためモデルが粗くても自然に見え，さらに不要なレンダリング処理より移動速度が落ちることもないことが確認された。

4. LODによるアニメーション

LODを利用することにより，カメラからの距離に応じて表示させる物体を変化させることができる。この本来の目的は，前述の通り，透視投影カメラで観測するとき，遠くに小さく見える物体を粗く定義することにより，レンダリング処理時間を軽減することである。

しかし，LODの機能は透視投影カメラのみならず，平行投影カメラを使用した場合でも有効である。平行投影の場合，奥行き情報が無視されるため，カメラの前進・後退によるスケール変化は起こらない。



(a)　レベル3　　(b)　レベル4　　(c)　レベル5

図2　ボリュームデータの可視化における Level of Detail

したがって，平行投影カメラとLODノードとを組み合わせ，z軸（奥行き）方向を時間軸と考えると，理論上では無限長のアニメーションが実現できることになる。ここで，アニメーションの各フレームに対応するものは，3次元の物体である。

ただし，平行投影カメラで観測する場合，カメラからの距離に応じたスケーリングがなされないため，あらかじめ可視化すべきデータを一定範囲内に正規化しておかねばならない。この範囲はVRMLブラウザに依存するかもしれないが，今回の実験では，

$$\begin{cases} -0.5 \leq x \leq 0.5 \\ -0.5 \leq y \leq 0.5 \end{cases} \tag{2}$$

となるように正規化して表示を行った。

4．1　3次元形状のアニメーション

3次元形状のアニメーションの例として，まずローレンツアトラクタ（Lorenz attractor）を対象とした。図3(a)は，ローレンツ方程式

$$\begin{cases} \dfrac{dX}{dt} = \mathrm{Pr}Y - \mathrm{Pr}X \\ \dfrac{dY}{dt} = -XZ + RX - Y \\ \dfrac{dZ}{dt} = XY - BZ \end{cases} \tag{3}$$

において，各係数を Pr ＝10，R ＝28，B ＝8/3とおき，時間方向の刻み $\Delta t = 0.01$ として生成された3次元の時系列データを順に線分で結ぶことにより描いたものである。

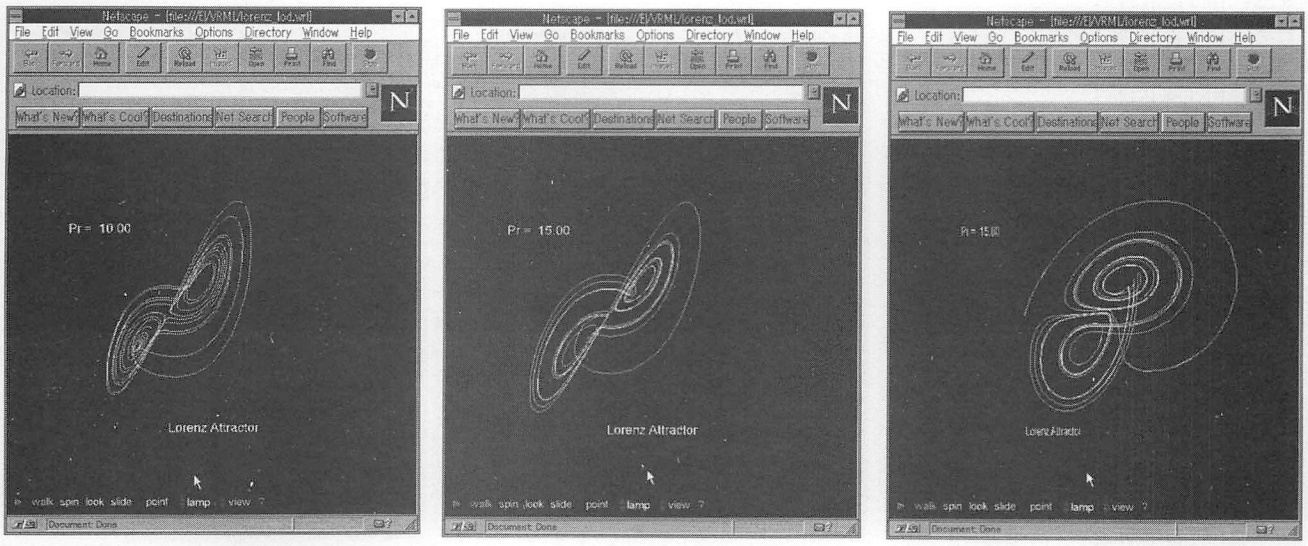
線分の描画は，リスト4に示すように，Coordinate3ノードとIndexedLineSetノードにより行う。

リスト4　IndexedLineSetノードの書式

```
Coordinate3 {
  point [
    x1, y1, z1,
    x2, y2, z2,
        ...
    xn, yn, zn
  ]
}
IndexedLineSet {
  coordIndex [
    0, 1, ... , n-1, -1
  ]
}
```

ここでは，係数Prの変化によるローレンツアトラクタの形状変化をアニメーションとして観察した。図3(b)がPr＝15としたときのアトラクタ形状である。図3(c)は(b)の状態からアトラクタを右方向へ45度回転させたものである。

次に，Takensの埋込み定理により状態空間に再構成された音声データのアトラクタを3次元表示し，話者による変化をアニメーションとして観察した。図4(a)が話者Aの母音「い」の再構成アトラクタ，図4(b)が話者Bの母音「い」の再



(a) Pr＝10, R＝28, B＝8/3　(b) PR＝15, R＝28, B＝8/3　(c) (b)を右45度回転

図3　ローレンツアトラクタのアニメーション

(a) 話者Aの母音「い」　(b) 話者Bの母音「い」

図4　再構成アトラクタのアニメーション

構成アトラクタである。

このように，パラメータの変化に応じた3次元形状の変化が，アニメーションにより理解しやすくなる。

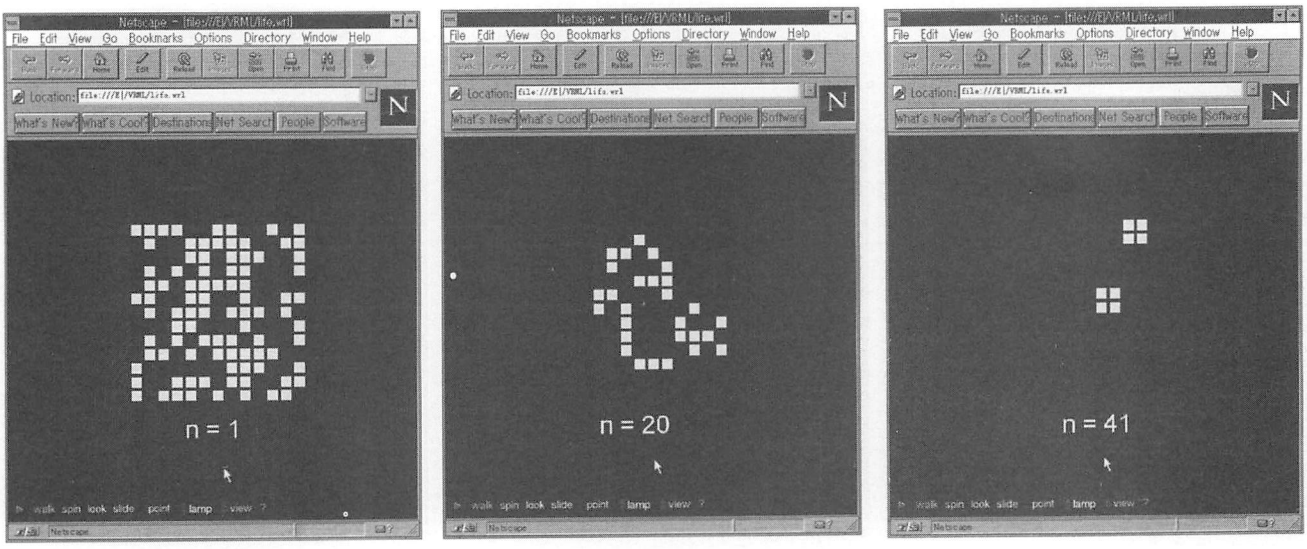
4．2　2次元形状のアニメーション

3次元形状のアニメーションと同様に，通常の2次元形状のアニメーションも容易に実現できることを示す。

まず，Conwayのライフゲームの世代進行に伴うパターン変化をアニメーションで示した。図5における白色の正方形が生きているセルを表しており，(a)でランダムに与えた初期パターンが，世代の進行にしたがって変化し，(c)の第41世代で収束する様子がわかる。

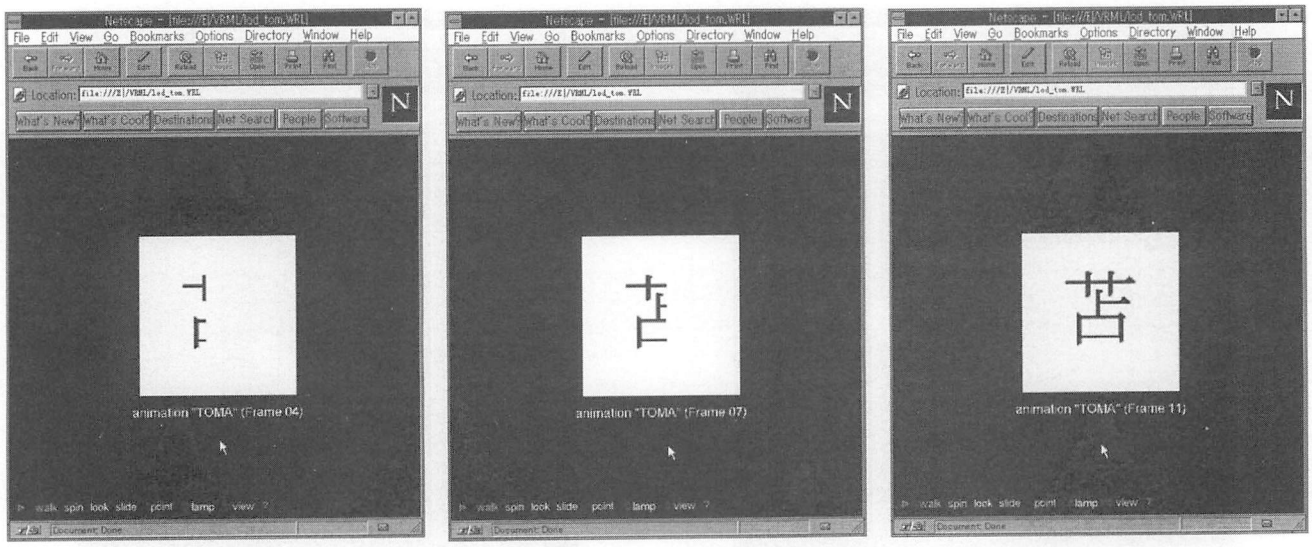
白色の正方形は，リスト5に示すように，Coordinate3ノードとIndexedFaceSetノードによって描画されている。

次に，面にマッピングされた画像によるアニメーションを行った。図6には，全12フレームによる「苦」の文字が現れるアニメーションの一部を示した。



(a)　第1世代　　(b)　第20世代　　(c)　第41世代

図5　ライフゲームのアニメーション



(a)　フレームNo04　　(b)　フレームNo07　　(c)　フレームNo11

図6　画像マッピングによるアニメーション

リスト5　IndexedFaceSetノードの書式

```
Coordinate3 {
  point [
    x1, y1, z1,
    x2, y2, z2,
    x3, y3, z3,
    x4, y4, z4
  ]
}
IndexedFaceSet {
  coordIndex [
    0, 1, 2, 3, -1
  ]
}
```

ここでは，IndexedFaceSetノードにより定義された面に対し，リスト6に示すTexture2ノードにより画像をマッピングした。filenameフィールドに記述するファイル名によりマッピングされる画像を指定する。

リスト6　Texture2ノードの書式

```
Texture2 {
  filename "filename"
}
```

5．形状ノードの再利用による簡易モデリング

定義済みの物体を再利用するためのUSE命令を使うと，基になる物体のアフィン変換による軌跡を利用したモデリングが容易に行える。

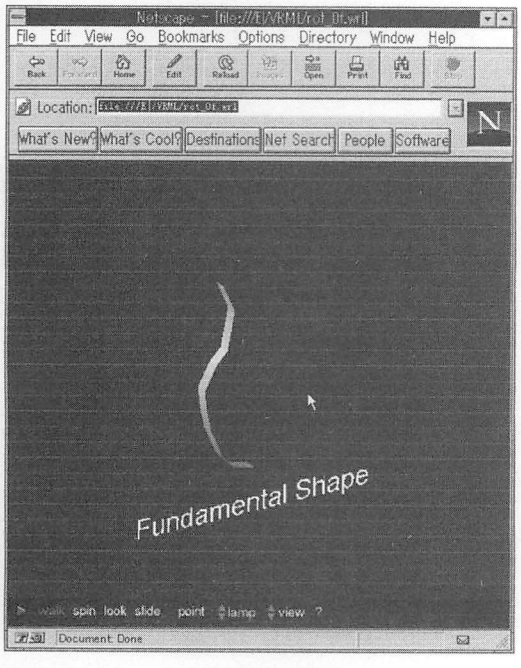
例えば，回転移動スイープによる回転体のモデリングや，アフィン変換に縮小を取り入れることによる3次元フラクタル形状のモデリングなどが挙げられる。

ここでは，回転体のモデリング例を示す。まず，図7(a)に示す基本形状（3次元）を定義し，これをy軸のまわりに回転させていったときの軌跡により図7(b)の回転体をモデリングした

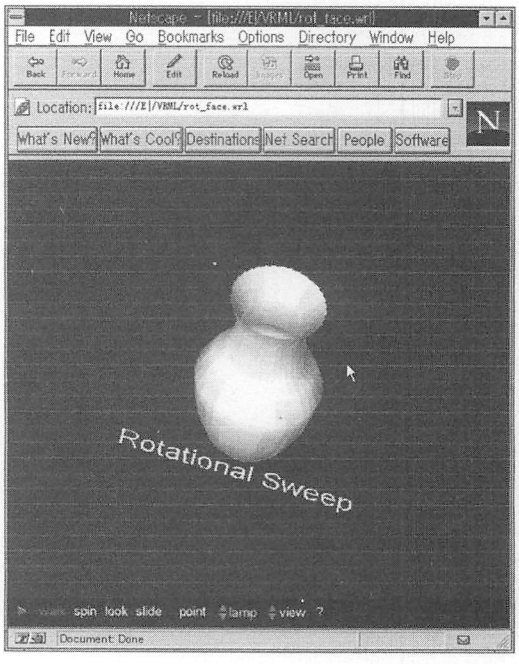
図7(b)の回転体の定義部分を出力するためには，リスト7のC言語ソースプログラムで十分である。ここで，LINESとは図7(a)に示す基本形状につけられた名前であり，Nは円周方向の分割数であり36とした。

リスト7　回転体の定義の出力用プログラム例

```
dr = 2.0 * M_PI / N;
for (i = 0; i < N; i ++) {
 printf("\tTransform {\n");
 printf("\t\trotation 0 1 0 %lf\n", dr);
 printf("\t}\n");
 printf("\tUSE LINES\n");
}
```



(a)　基本形状



(b)　生成された回転体

図7　回転体のモデリング例

6．おわりに

3次元シーンの記述およびブラウジングの容易性からVRML1.0に着目し，いくつかの実験を行うことにより身近なデータの可視化ツールとしての有効性を示した。

まず，LODの構成例として，視点からの距離に応じたボリュームデータの表示形状を，ボクセルのOctreeの表現から生成する実験を行った。

次に，LODと平行投影カメラを併用することにより，VRML1.0でも3次元や2次元のアニメーションを実現できることを示した。

さらに，USE命令を用いた形状ノードの再利用により擬似的なスイープが可能であることを，回転体の生成例で示した。

VRMLは現在も発展を続けている言語であり

,2.0 仕様では behavior の記述，仮想空間の共有，音声・動画のサポートなど数々の新機能が追加され，ますます多様の応用可能性が考えられるだろう。例えば，今回の実験ではボリュームデータを Level of Detail 表示したにすぎないが，VRML2.0 を利用すれば Java により behavior を記述することにより，マウスを用いたインタラクティブなボリュームデータのモデリングシステムを作ることができる[4)]。

参考文献

1) G. Bell, A. Parisi, M. Pesce : "The Virtual Reality Modeling Language Version 1.0 Specification", http://vag. vrml. org/vrml10c. html (1995)

2) M. Pesce :"VRML を知る"，トッパン (1996)

3) 次世代 www 研究会："web 進化論"，オーム社 (1996)

4) 佐々木あゆみ："VRML によるボリュームモデリング"，苫小牧工業高等専門学校準学士論文 (1997)

(平成8年11月29日受理)