ヒストリベースパラメトリックモデルにおける形状生成プロセスの等価性証明に関する研究

菊　地　慶　仁

Research for equivalence proofing of 3-D shape formation process on history based parametric model

Yoshihito KIKUCHI

概　要

ヒストリベースパラメトリックモデルでは，同じプログラムによって異なるシステム上で生成される形状モデルが同じ形状であるかどうかの証明が基本的な問題となる。本報告は C.A.R. Hore によって提唱された公理系に基づくプログラムの結果の証明方法を形状定義に適応する方式を提案し，PC 上で稼動する 3 次元 CAD を用いて提案した方式の有効性を確認する。

1．ヒストリベースパラメトリックモデルにおける問題点

3 次元の形状モデラを用いて生成された形状データを保存する方法として，

1．形状を定義するデータを保存する方法

2．形状を生成する手順をデータとして保存する方法

の 2 つがあり，後者はヒストリベースパラメトリックモデルと呼ばれている。この方式は，CAD システムに対するキーボード及びマウスなどのポインティングデバイスからの入力を記録しておく従来のマクロなどと同じ方式である。CAD システムへの操作入力をファイルとして保存したものを以下ではプログラムと呼ぶ。図 1 にシステムに対する入力，プログラム，及び実際のデータの関係を示す。

この形式では，次のような特徴がある。

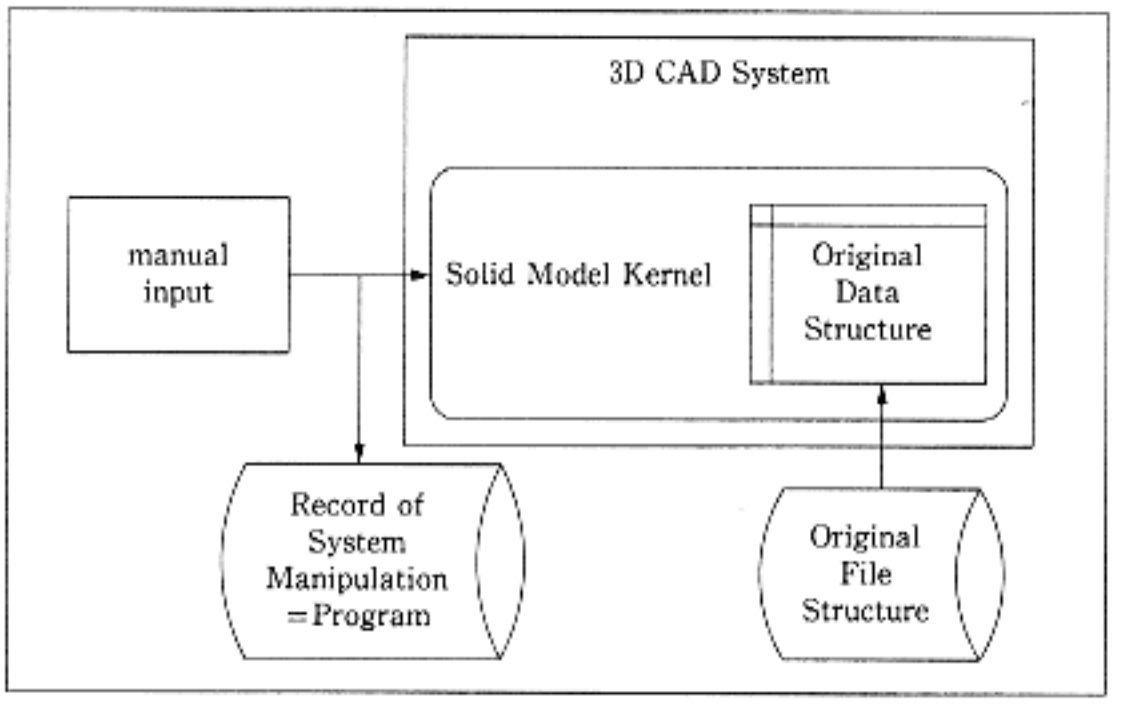
・それぞれのシステムの内部データ構造が違っている場合でも，複数のシステムでの操作が統一化され，その操作に対する対応が各モデリングシステムで共通化されていれば，作り出されるモデルは同じ形状になると考えることができる。即ち，システムの内部データ構造やモデルの意味を STEP [2] [3] のように厳密に共通化していなくてもデータの共有や交換が行える可能性がある。

・特定の操作中に入力されたパラメータを，後から変更してモデルの再構築を実行させることによって，パラメトリックモデルとして活用できる可能性がある。

しかしながら，ヒストリベースパラメトリックモデルは，実用化の段階には達していない。その理由として次のような問題点が上げられる。

・一般にCADシステムの内部データの構造は公開されておらずに，ブラックボックスとして扱われている。プログラムのステートメント一つ一つを解析したとしても，システム内部でどういった処理が行われているかを正確に表現することはできず，複数システムによって等価なモデルが作られるという保証を行うことが出来ない。また，その検証手段が確立されていない。

図1　入力操作を記録したプログラムとデータの関係



・結果として異なったモデルが生成された判断するための根拠，あるいはどこまでを許容して等価／非等価に判断するか，といった議論を行うための基盤が必要となるが，このような判断基準が確立されていない。

・上記の議論の根拠となる方法論が存在しない場合の代替案として，標準的なプログラムを用意し，それぞれのシステムでの生成結果を比較する方式が考えられる。しかしながら，そのプログラムが必要十分であることを保証することが出来ない。

・実際の作業をプログラムとして記録した場合，試行錯誤的に行なわれた作業によってプログラムが冗長となることが考えられる。この冗長性を省いて等価な結果が得られるプログラムを生成する必要がある。

・プログラム中では，形状プリミティブの生成及び消去，グループ化及びグループ解除，等が繰り返し行なわれる。操作プログラムをパラメトリックモデルの定義として考えた場合

に，このプログラム中で，特定の操作における引数の変更が，プログラムのどの範囲に影響を及ぼすか，あるいは意図した結果を十分に得られるかどうかを明確に求めることが出来ない。

上記の問題点を解決し，ヒストリベースパラメトリックモデルの実用化を進めるためには，まず入力プログラムから生成される同一システム内での形状モデルが等価であるかどうかを判定する基本的な方法を確立し，その方法論に基づいた形で

・同一プログラムを入力した場合の，複数のシステム内部挙動と結果となるモデルの評価

・プログラム中でパラメータを変更した際の影響範囲の推定

・プログラムの冗長性の排除

・複数システムでのモデルの等価性の保証。

などの技術開発の必要性が考えられる。

本報告では，以下の内容について報告を行う予定である。

・研究の背景及び問題点に関する分析

・Hore 論理の基本的な概念

・ヒストリベースパラメトリックモデルでの応用方式の提案

・2+1/2 次元レベルの形状モデルでの基本的な例題の証明

以下では，この問題の中核となるプログラムの等価性証明について述べる。

2．本研究で用いる方法論

Hore 論理は，プログラム言語 PASCAL の意味論的な仕様を公理として定義し，各プログラムステートメントからの結果を推定するために用いられた。その応用として，グラフィックディスプレイ上での 2 次元の図形描画が等価かどうかの判定を行った研究 [1] があるが，本研究のように 3 次元形状モデルを対象とした報告例はまだない。評価は以下の手順で行う（図 2）。

1．CAD システム内部（通常はソリッドモデルカーネルなどと呼ばれる）で実行される各処理に相当する公理を設ける。具体的には，

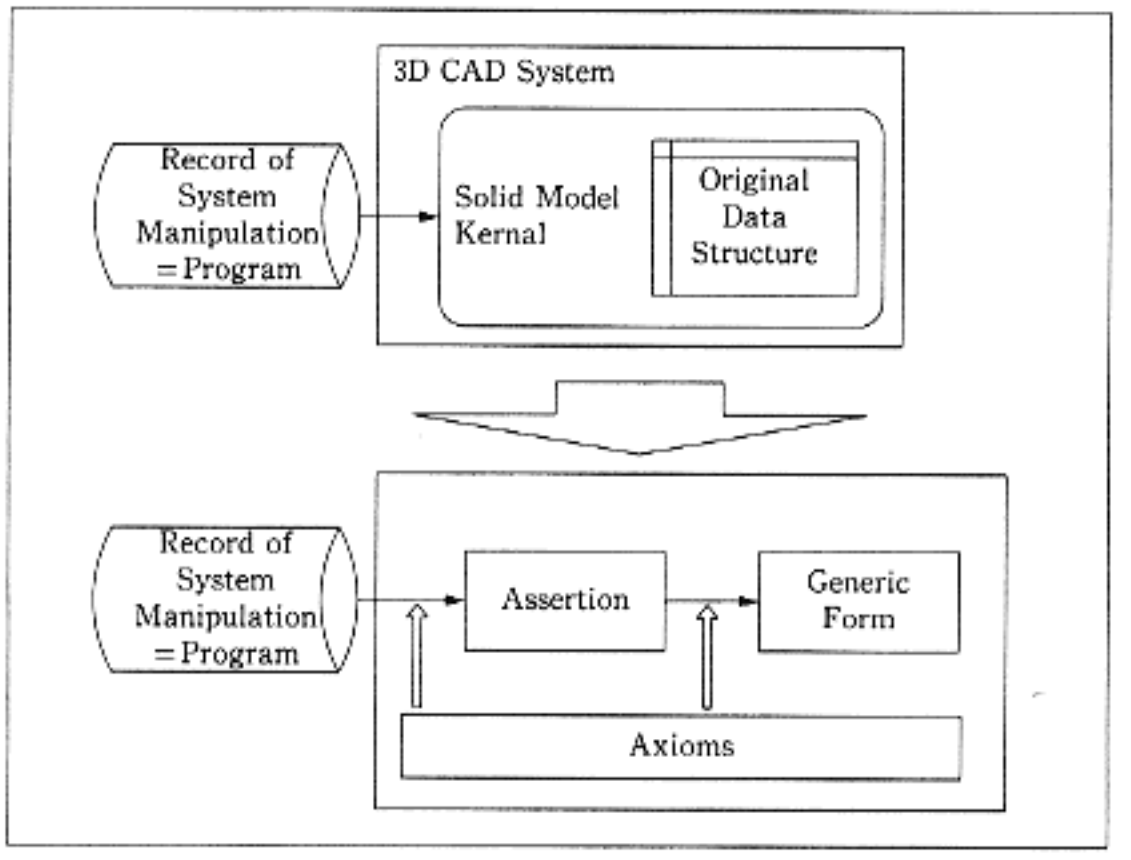
(a)プログラム開始時の初期化の設定

(b)プログラムを実行する度の形状要素等の内部変数の生成や変更などを表現する。

などを行なう。こうすることによって，CAD システムの内部メカニズムの挙動に関して数学的な取り扱いを可能とさせる。

2．入力されるプログラムを公理を用いた“表明（Assertion）”に変換する。表明はプログラムに一対一に対応することになる。さらに表明を変換用の公理を用いて標準形（Generic Form）と呼ばれる形式に変形する。

図2　ソリッドモデル中でのプログラムの実行と Hore 論理による公理系の対応



3．複数のプログラムを，それぞれ標準形に変形し，その標準形の中でのパラメータが何処で異なっているかの比較から等価かどうかの判定を行う。

ただし，実際のモデリングシステムのカーネル及びデータ構造の全てが Hore 論理における表明及び標準形に１対１に対応しているわけではなく，実際の描画などに関係の無い部分は省略されている。

この方式を用いる際のメリットとしては，以下の項目を挙げることができる。

・CAD システムが標準的な公理系をサポートしていることが保証できれば，別々のシステムであっても，プログラムの結果として得られる形状モデルが等価であることを証明できる。特定の公理系を基準となる規格としてヒストリベースパラメトリックモデルの標準化の可能性がある。

・等価ではないと判定するた場合でも，どこの部分で非等価の判定を行ったかは残せるので，議論の土台を得ることができる。

3．プログラム証明実験

3．1　実験の目的と方法について

本報告では，提案した方式の有効性を確認するために，単純な形状での証明問題を試みる。具体的には，直方体の平行移動と回転移動を用いたプログラム証明を行なう。プログラム証明

の過程は，PC 用 CAD の VectorWorks を用いて行なった。始めに，ある形状を生成するプログラム 1 を作成し，同じプログラムのパラメータを変えて，結果的に同じ形状を得るプログラム 2 を作成する。始めに VectorWorks 上で 2 つのプログラムの結果の形状が一致することを確認しておく。その上でプログラムの表明，公理を導き，プログラム 1 とプログラム 2 を Hore の論理を用いて標準形で表わし，その 2 つの形状が等しいことを証明する。

本報告での形状生成は，始めは木構造の 3 次元形状に対するオペレーションとして扱い（以下木構造画像），これを，厚みと領域とを持った画像（以下連続画像）へと公理を用いて変換して表明を導出する。さらに連続画像上での変換用の公理を用いて一般形へと変換して等価性を判定する。領域の等価は，底面の領域と高さの組み合わせで判定する。ただし，プログラム中の回転移動を伴う操作を組み合わせたプログラムを扱うので，形状は直方体のみとして，領域と高さは直方体の 8 頂点の座標によって表現することとする。

3．2　形状生成プログラム

本報告で証明実験に用いるプログラムを以下に示す。

プログラム 1

```
BEGIN
NameClass('ippan') ;
BeginXtrd(0 mm,50 mm) ;
PenSize(1) ;
PenPad(2) ;
FillPad(1) ;
Rect(0 mm,40 mm,30 mm,0 mm) ;
EndXtrd ;
Rotate 3 D(#0.0'0",#0.0'0",#0.0'0") ;
Move 3 D(0 mm,0 mm,0 mm) ;
END.
```

プログラム 2

```
BEGIN
NameClass('ippan') ;
BeginXtrd(0 mm,40 mm) ;
PenSize(1) ;
PenPad(2) ;
FillPad(1) ;
Rect(−30 mm,0 mm,0 mm,50 mm) ;
EndXtrd ;
Rotate 3 D(#270.0'0",#0.0'0",#0.0'0") ;
Move 3 D(30 mm,0 mm,0 mm) ;
END.
```

1．プログラム 1

1 行目の “NameClass('ippan')” で，図形のクラスを示している。2 行目の “BeginXtrd (0 mm,50 mm)” から 7 行目の “EndXtrd” で，直方体の上と下の面（蓋）の高さに基づいた立体を定義している。3 行目の“PenSize(1)”，4 行目の“PenPat(2)”，5 行目の“FillPat(1)”では，線の太さ，線の種類，図形のファイルパターンを定義している。6 行目の“Rect(0 mm,40 mm,30 mm,0 mm)”では，3 次元の形状を造るために 2 次元の画像（長方形）を

造るもので，その長方形のxy平面上での左上の点と右下の点のxy座標を示している。ここまでででx軸に30 mm，y軸に40 mm，z軸に50 mmの直方体が生成される。

8行目の“Rotate 3 D(#0.0'0",#0.0'0",#0.0'0")”では，各xyz軸で回転させる角度を示している。そして，9行目の“Move 3 D(0,0,0)”で，各xyz軸方向への移動を定義する。ただし実際には，移動量，回転角ともに0なので，作成された形状は，そのままの位置に留まることになる。

2．プログラム2

プログラム1の各ステートメントにおける引数を変更して，同じ形状を描くプログラム2を作成する。プログラム2では，x軸に−30 mm，y軸に50 mm，z軸に40 mmの直方体を造り，“Rotate 3 D(#270'0'0,#0'0'0,#0'0'0)”でx軸に対し270度回転させ，“Move 3 D(30 mm,0,0)”でx方向へ30 mm移動させる。

3．3　プログラムからの表明の導出のための公理

始めに木構造画面から連続画面系へと変換して表明を導出するための公理を示す。これらの公理は，証明を行なうために必要最低限な変数の挙動に関してのみ取り扱っているので，証明に直接関係の無い変数の挙動は公理としては扱っていない。

$tp0.$ $display(Begin)=display(nullpict)$

$curobj(Begin)=nullpict$

$tp1.$ $curtop(BeginXtrd(A,B))=B$

$curbottom(BeginXtrd(A,B))=A$

$curthic(BeginXtrd(A,B))=B-A$

$tp2.$ $display(Rect(x_a,y_a,x_b,y_b,curbottom,curtop))$

$=p.sum(curdisp,\ display(octv(x_a,y_a,x_b,y_b,curbottom,curtop)))$

$curobj(Rect(x_a,y_a,x_b,y_b,curbottom,curtop))$

$=octv(x_a,y_a,x_b,y_b,curbottom,curtop)$

$tp3.$ $display(Rotate\ 3D(curobj,\theta_x,\theta_y,\theta_z))$

$=p.sum(curdisplay,\ deldisplay(curobj),display(T^r(\theta_x,\theta_y,\theta_z)\times curobj))$

$curobj(Rotate\ 3D(curobj,\theta_x,\theta_y,\theta_z))$

$=octv(T^r(\theta_x,\theta_y,\theta_z)\times curobj)$

$tp4.$ $display(Move\,3D(curobj,\ \delta x,\ \delta y,\ \delta z))$

$=p.sum(curdisplay,\ deldisplay(curobj),display(T^m(\delta x,\ \delta y,\ \delta z)\times curbnj))$

$curobj(Move\,3D(curobj,\ \delta x,\ \delta y,\ \delta z))$

$=octv(T^m(\delta x,\ \delta y,\ \delta z)\times curobj)$

*tp*0.は，プログラムの実行が開始された時点での各変数の初期化に対応している。ここでは，プログラムBeginを入力として，変数の初期化$display(Begin)=display(nullpict)$及び$curobj(Begin)=nullpict$を行なっている。

*tp*1.～*tp*4.は，木構造型画像への操作毎に対応している。プログラムの各ステートメントを引数とし，連続画像上での変数や，連続画像上での表現(目に見える様子)を規定する[1]。

以下で，主な変数及び関数について述べる。

・各プログラムステートメントに対する*display*の公理は，木構造画像型のオペレーションを引数にとり，連続画像上での表現を返す形になっている。結果の値は*curdisplay*の値に保存される。*curdisplay*は，プログラムの各ステートメントを実行する時点で表示される連続画像オブジェクトである。この値は次の*operation*を*display*する際に引数としても使われる。すなわち，毎に各ステートメントこの公理を適用することによって，木構造画像に対しての操作に対する連続画像上での結果が得られることになる。

・*p.sum*は，画像の重ね合わせを行い結果を返す関数である。

・*curthic*は，その時点の連続画像の厚さ(面の厚さ)を導く公理である。

・*curobj*は，その時点で形状操作の対象となるオブジェクトを示す。

・*octv*は，底面の2端点と厚さ，又は連続画像オブジェクト*curobj*と座標変換用パラメータを引数として領域を返す関数で，その領域は同次座標表現された8頂点の集合で定義される。頂点のZ座標値は，その前に呼ばれた$BeginXtrd(A,B)$のAとBの値で設定された*curtop*と*curbottom*から決定される。

$$octv(x_a, y_a, x_b, y_b, curbottom, curtop)=$$

$$\begin{bmatrix} x_a & x_a & x_a & x_a & x_b & x_b & x_b & x_b \\ y_a & y_a & y_b & y_b & y_a & y_a & y_b & y_b \\ curbottom & curtop & curbottom & curtop & curbottom & curtop & curbottom & curtop \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

・*deldisplay*は，指定された連続画像型の領域を消去する。

$Rotate\,3D(curobj, \theta_x, \theta_y, \theta_z)$及び$Move\,3D(curobj, \delta x, \delta y, \delta z)$によって頂点集合に適用される座標変換$T$を，回転座標変換$T^r$と並行移動$T^m$とに分けて次の形で定義する。

$$T = T^m(\delta x, \delta y, \delta z) \times T^r(\theta_x, \theta_y, \theta_z) \tag{1}$$

---

[1]オリジナルのプログラムでは，Rotate 3D(curobj, θ_x, θ_y, θ_z)及びMove 3D(curobj, δx, δy, δz)は，「その時点で(選択されている)オブジェクトに対して」であるとの暗黙の了解の元に実施されていた。すなわち，本来は明示的に引数とすべき「現在のオブジェクト*curobj*」が省略されている。本報告では，公理の適用による機械的な証明システムの開発も目的としているため，明示的な引数として示す形をとった。

$$T^m(\delta x, \delta y, \delta z)=\begin{pmatrix} 1 & 0 & 0 & \delta x \\ 0 & 1 & 0 & \delta y \\ 0 & 0 & 1 & \delta z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2}$$

$$T^r(\theta_x, \theta_y, \theta_z)=T^r_x(\theta_x)\times T^r_y(\theta_y)\times T^r_z(\theta_z) \tag{3}$$

$$T^r_x(\theta_x)=\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta_x & -\sin\theta_x & 0 \\ 0 & \sin\theta_x & \cos\theta_x & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4}$$

$$T^r_y(\theta_y)=\begin{pmatrix} \cos\theta_y & 0 & \sin\theta_y & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta_y & 0 & \cos\theta_y & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5}$$

$$T^r_z(\theta_z)=\begin{pmatrix} \cos\theta_z & -\sin\theta_z & 0 & 0 \\ \sin\theta_z & \cos\theta_z & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{6}$$

3．4　標準形の導出のための公理

以下に，標準形を導出するために使用する公理を定義する。

$p1.$　$p.display(P)=inspatch(nullpict, R^s(P))$

$p2.$　$p.deldisplay(P)=inspatch(nullpict, R^d(P))$

$p3.$　$p_1+inspatch(p_2, R)=addpatch(p_1, R)+restriction(p_2, \approx R)$

$p4.$　$restriction(nullpict, \approx R)=nullpict$

$p5.$　$p+nullpict=p$

$p6.$　$addpatch(nullpict, R)=inspatch(nullpict, R)$

$p7.$　$inspatch(nullpict, R^s)+inspatch(nullpict, R^d)=nullpict$

$p8.$　$(inspatch(nullpict, R^s)+inspatch(nullpict, R^d))$
$=inspatch(nullpict, R^s)+inspatch(nullpict, R^d)$

ここでR^s及びR^dは，その時点の画像Pもしくは描画プログラムのパラメータを与えることによって定義される領域で，画像の生成及び消去に対応している。すなわちR^dは，消去されるべき領域を示している。

$$R^s(octv(x_a, y_a, x_b, y_b, curbottom, curtop))$$

$$= rg.solid\begin{bmatrix} x_a & x_a & x_a & x_a & x_b & x_b & x_b & x_b \\ y_a & y_a & y_b & y_b & y_a & y_a & y_b & y_b \\ curbottom & curtop & curbottom & curtop & curbottom & curtop & curbottom & curtop \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{7}$$

$$R^d(octv(x_a, y_a, x_b, y_b, curbottom, curtop))$$

$$= rg.delsolid\begin{bmatrix} x_a & x_a & x_a & x_a & x_b & x_b & x_b & x_b \\ y_a & y_a & y_b & y_b & y_a & y_a & y_b & y_b \\ curbottom & curtop & curbottom & curtop & curbottom & curtop & curbottom & curtop \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{8}$$

3．5　頂点座標及び座標変換

プログラム1で生成される立方体の8頂点の集合は，端点の座標(0,40)と(30,0)，及び*BeginXtrd*(0,50)から次となる。

$$octv(0,40,30,0,0,50)=\begin{bmatrix} 0 & 0 & 0 & 0 & 30 & 30 & 30 & 30 \\ 40 & 40 & 0 & 0 & 40 & 40 & 00 & 00 \\ 0 & 50 & 0 & 50 & 0 & 50 & 0 & 50 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{9}$$

また，プログラム1での座標変換は以下の2つの式となる。

$$T^r(0,0,0)=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{10}$$

$$T^m(0,0,0)=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{11}$$

3．6　プログラム1に対応する表明の導出

プログラム1に対応する表明P_{one}を導出する。NameClass('ippan'), PenSize(1), PenPat(2), FillPat(1)は，今回の等価性の証明とは直接関係していないので省略する。

Begin(公理 tp 0) より

$tp0.\quad display(Begin)=display(nullpict)$

$curobj(Begin)=nullpict$

BeginXtrd(0 mm, 50 mm) (公理 tp 1) より

$tp1.\quad curtop(BeginXtrd(A,B))=B$

$curbottom(BeginXtrd(A,B))=A$

$curthic(BeginXtrd(A,B))=B-A$

$curtop(BeginXtrd(0mm,50mm))=50mm$

$curbottom(BeginXtrd(0mm,50mm))=0mm$

$curthic(BeginXtrd(0mm,50mm))=50mm$

Rect(0 mm, 40 mm, 30 mm, 0 mm) (公理 tp 2) より

$tp2.\quad display(Rect(x_a, y_a, x_b, y_b, curbottom, curtop))$

$=p.sum(curdisp,\ display(octv(x_a, y_a, x_b, y_b, curbottom, curtop)))$

$curobj(Rect(x_a, y_a, x_b, y_b, curbottom, curtop))$

$=octv(x_a, y_a, x_b, y_b, curbottom, curtop)$

$display(Rect(0mm, 40mm, 30mm, 0mm, 0mm, 50mm)$

$=display(p.nullpict)+display(octv(0mm, 40mm, 30mm, 0mm, 0mm, 50mm))$ (12)

$curobj(Rect(0mm, 40mm, 30mm, 0mm, 0mm, 50mm))=octv(0mm, 40mm, 30mm, 0mm, 0mm, 50mm)$ (13)

Rotate 3 D(#0.0'0", #0.0'0", #0.0'0") ; (公理 tp 3) より

$tp3.\quad display(Rotate\,3D(curobj, \theta_x, \theta_y, \theta_z)$

$=p.sum(curdisplay, deldisplay(curobj),\ display(T^r(\theta_x, \theta_y, \theta_z)\times curobj))$

$curobj(Rotate\,3D(curobj, \theta_x, \theta_y, \theta_z))$

$=octv(T^r(\theta_x, \theta_y, \theta_z)\times curobj)$

$P_{one}=$

$display(Rotate\,3D(\#0.0'0", \#0.0'0", \#0.0'0"))=$

$nullpict$

$+display(octv(0mm, 40mm, 30mm, 0mm, 0mm, 50mm))$

$+deldisplay(octv(0mm, 40mm, 30mm, 0mm, 0mm, 50mm))$

$+display(T^r(\#0.0'0", \#0.0'0", \#0.0'0")\times octv(0mm, 40mm, 30mm, 0mm, 0mm, 50mm))$ (14)

$$curobj = T^{r}(\#0.0'0'', \#0.0'0'', \#0.0'0'') \times octv(0mm, 40mm, 30mm, 0mm, 0mm, 50mm) \quad (15)$$

Move 3 D(0 mm, 0 mm, 0 mm) ; (公理 tp 4)

$tp4.$ $display(Move3D(curpbj, \delta x, \delta y, \delta z))$

$= p.sum(curdisplay, deldisplay(curobj), display(T^{m}(\delta x, \delta y, \delta z) \times curbnj))$

$curobj(Move3D(curobj, \delta x, \delta y, \delta z))$

$= octv(T^{m}(\delta x, \delta y, \delta z) \times curobj)$

$P_{one} =$

$display(Move3D(0mm, 0mm, 0mm)) =$

$nullpict$

$+ display(octv(0mm, 40mm, 30mm, 0mm, 0mm, 50mm))$

$+ deldisplay(octv(0mm, 40mm, 30mm, 0mm, 0mm, 50mm))$

$+ display(T^{r}(\#0.0'0'', \#0.0'0'', \#0.0'0'') \times octv(0mm, 40mm, 30mm, 0mm, 0mm, 50mm))$

$+ deldisplay(T^{r}(\#0.0'0'', \#0.0'0'', \#0.0'0'') \times octv(0mm, 40mm, 30mm, 0mm, 0mm, 50mm))$

$+ display($

$T^{m}(0, 0, 0) \times T^{r}(\#0.0'0'', \#0.0'0'', \#0.0'0'') \times octv(0mm, 40mm, 30mm, 0mm, 0mm, 50mm))$

(16)

$curobj(Move3D(0mm, 0mm, 0mm)) =$

$T^{m}(0, 0, 0) \times T^{r}(\#0.0'0'', \#0.0'0'', \#0.0'0'') \times octv(0mm, 40mm, 30mm, 0mm, 0mm, 50mm)$

(17)

3．7　プログラム 2 に対する表明の導出

プログラム 2 に対する表明P_{two}を求める。プログラム 2 はプログラム 1 と同じ手順だが次の 3 点が異なっている。

1．最初に作られる立方体が異なる。$A=-40$，$B=0$，$Rect(-30, 0, 0, 50)$

2．x 軸周りで 270 度の回転座標変換を行なう

3．x 軸方向に 30 mm の並行移動座標変換を行なう

対応するパラメータを式(16)及び式(17)に代入して展開すると，

$P_{two} =$

$display(Move3D(30, 0, 0)) =$

$nullpict$

$+ display(octv(-30mm, 0mm, 0mm, 50mm, -40mm, 0mm))$

$$+deldisplay(octv(-30mm,0mm,0mm,50mm))$$
$$+display(T^{r}(\#270.0'0'',\#0.0'0'',\#0.0'0'')\times octv(-30mm,0mm,0mm,50mm,-40mm,0mm))$$
$$+deldisplay(T^{r}(\#270.0'0'',\#0.0'0'',\#0.0'0'')\times octv(-30mm,0mm,0mm,50mm,-40mm,0mm))$$
$$+display($$
$$T^{m}(30,0,0)\times T^{r}(\#270.0'0'',\#0.0'0'',\#0.0'0'')\times octv(-30mm,0mm,0mm,50mm,-40mm,0mm))$$

(18)

$$curobj(Move3D(30,0,0))=$$
$$T^{m}(30,0,0)\times T^{r}(\#270.0'0'',\#0.0'0'',\#0.0'0'')\times octv(-30mm,0mm,0mm,50mm,-40mm,0mm) \quad (19)$$

となる。

3．8 P_{one}の標準形による表現

表明P_{one}、式(16)を標準形で表わす。公理$p1$及び$p2$，上記の領域R^{s}及びR^{d}を用いて$p.display$と$p.deldisplay$を集合表現の領域型に書き換える。

$$P_{one}=$$
$$nullpict$$
$$+inspatch(nullpict,R^{s}(0mm,40mm,30mm,0mm))$$
$$+inspatch(nullpict,R^{d}(0mm,40mm,30mm,0mm))$$
$$+inspatch(nullpict,R^{s}(T^{r}(\#0.0'0'',\#0.0'0'',\#0.0'0'')\times octv(0mm,40mm,30mm,0mm,0mm,50mm))$$
$$+inspatch(nullpict,R^{d}(T^{r}(\#0.0'0'',\#0.0'0'',\#0.0'0'')\times octv(0mm,40mm,30mm,0mm,0mm,50mm))$$
$$+inspatch(nullpict,$$
$$R^{s}(T^{m}(0,0,0)\times T^{r}(\#0.0'0'',\#0.0'0'',\#0.0'0'')\times octv(0mm,40mm,30mm,0mm,50mm,0mm)) \quad (20)$$

公理 p 7 $inspatch(nullpict,R^{s}(P))+inspatch(nullpict,R^{d}(R))=nullpict$
を用いると

$$P_{one}=p.nullpict+inspatch(nullpict,$$
$$R^{s}(T^{m}(0,0,0)\times T^{r}(\#0.0'0'',\#0.0'0'',\#0.0'0'')\times octv(0mm,40mm,30mm,0mm,0mm,50mm))) \quad (21)$$

公理 p 3 を用いると，

$$P_{one}$$
$$=addpatch(p.nullpict,$$
$$R^{s}(T^{m}(0,0,0)\times T^{r}(\#0.0'0'',\#0.0'0'',\#0.0'0'')\times octv(0mm,40mm,30mm,0mm,0mm,50mm))$$
$$+restriction(nullpict,$$

$$\approx R^s(T^m(0,0,0)\times T^r(\#0.0'0'',\#0.0'0'',\#0.0'0'')\times octv(0mm,40mm,30mm,0mm,0mm,50mm))$$

これに公理ｐ4，ｐ5，ｐ6を用いると，

$$P_{one}=addpatch(p.nullpict,$$
$$R^s(T^m(0,0,0)\times T^r(\#0.0'0'',\#0.0'0'',\#0.0'0'')\times octv(0mm,40mm,30mm,0mm,0mm,50mm)))$$

(22)

から

$$P_{one}=inspatch(nullpict,$$
$$R^s(T^m(0,0,0)\times T^r(\#0.0'0'',\#0.0'0'',\#0.0'0'')\times octv(0mm,40mm,30mm,0mm,0mm,50mm)))$$

(23)

となる。

3．9　P_{two}の標準形による表現

次にP_{two}を標準形で表わす。公理 $p1$，$p2$を用いると，

$$P_{two}=$$
$$nullpict$$
$$+inspatch(nullpict,R^s(-30mm,0mm,0mm,50mm))$$
$$+inspatch(nullpict,R^d(-30mm,0mm,0mm,50mm)$$
$$+inspatch(nullpict,R^s(T^r(\#270.0'0'',\#0.0'0'',\#0.0'0'')\times octv(-30mm,0mm,0mm,50mm,-40mm,0mm))$$
$$+inspatch(nullpict,R^d(T^r(\#270.0'0'',\#0.0'0'',\#0.0'0''))\times octv(-30mm,0mm,0mm,50mm,-40mm,0mm))$$
$$+inspatch(nullpict,$$
$$R^s(T^m(30,0,0)\times T^r(\#270.0'0'',\#0.0'0'',\#0.0'0''))\times octv(-30mm,0mm,0mm,50mm,-40mm,0mm))$$

(24)

公理 $p7$，$p8$を用いると，

$$P_{two}=p.nullpict+inspatch(nullpict,$$
$$R^s(T^m(30,0,0)\times T^r(\#270.0'0'',\#0.0'0'',\#0.0'0'')\times octv(-30mm,0mm,0mm,50mm,-40mm,0mm))$$

(25)

公理ｐ3を用いると，

$$P_{two}=$$
$$addpatch(p.nullpict,$$

$$T^{m}(30,0,0)\times T^{r}(\#270.0'0'',\#0.0'0'',\#0.0'0'')\times octv(-30mm,0mm,0mmmm,50mm,-40mm,0mm))$$
$$+restriction(nullpict,$$
$$\approx R^{s}(T^{m}(30,0,0)\times T^{r}(\#270.0'0'',\#0.0'0'',\#0.0'0'')\times octv(-30mm,0mm,0mm,50mm,-40mm,0mm) \quad (26)$$

これに公理ｐ4，ｐ5を用いると，

$$P_{two}=$$
$$addpatch(p.nullpict,$$
$$R^{s}(T^{m}(30,0,0)\times T^{r}(\#270.0'0'',\#0.0'0'',\#0.0'0'')\times octv(-30mm,0mm,0mm,50mm,-40mm,0mm)) \quad (27)$$

となり

$$P_{two}=$$
$$inspatch(nullpict,$$
$$R^{s}(T^{m}(30,0,0)\times T^{r}(\#270.0'0'',\#0.0'0'',\#0.0'0'')\times octv(-30mm,0mm,0mm,50mm,-40mm,0mm)) \quad (28)$$

となる。

3.10 プログラム1とプログラム2の公理的な証明

前節からP_{one}，P_{two}は式(23)(28)となる。P_{one}とP_{two}は，ともに最終的に“$nullpict$”による空の状態から，“$rg.solid$”で得られる領域を描くものである。P_{one}，P_{two}を定義する頂点列をX_{one}及びX_{two}とすると

$$X_{one}$$
$$=T_{m}(0,0,0)\times T^{r}(\#0.0'0'',\#0.0'0'',\#0.0'0'')\times octv(0mm,40mm,30mm,0mm,0mm,50mm))$$
$$=\begin{pmatrix} 0 & 0 & 0 & 0 & 30 & 30 & 30 & 30 \\ 40 & 40 & 0 & 0 & 40 & 40 & 00 & 00 \\ 0 & 50 & 0 & 50 & 0 & 50 & 0 & 50 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \quad (29)$$

$$X_{two}$$
$$=T^{m}(30,0,0)\times T^{r}(\#270.0'0'',\#0.0'0'',\#0.0'0'')\times octv(-30mm,0mm,0mm,50mm,-40mm,0mm)$$

$$
= \begin{bmatrix} 1 & 0 & 0 & 30 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -30 & -30 & -30 & -30 & -30 & 0 & 0 & -30 \\ 0 & 0 & 50 & 50 & 0 & 0 & 50 & 50 \\ -40 & 0 & -40 & 0 & -40 & 0 & -40 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}
$$

$$
= \begin{bmatrix} 1 & 0 & 0 & 30 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -30 & 0 & 0 & -30 & -30 & 0 & 0 & -30 \\ 40 & 0 & 40 & 0 & 40 & 0 & 40 & 0 \\ 0 & 0 & 50 & 50 & 0 & 0 & 50 & 50 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}
$$

$$
= \begin{bmatrix} 0 & 30 & 30 & 0 & 0 & 30 & 30 & 0 \\ 40 & 0 & 40 & 0 & 40 & 0 & 40 & 0 \\ 0 & 0 & 50 & 50 & 0 & 0 & 50 & 50 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}
$$

が得られる。X_{one}とX_{two}は頂点の出現順番は異なっているが同じ領域を示している。従ってP_{one}とP_{two}は等価になる。以上で二つのプログラムの等価性が証明された。

4．結　論

本報告では，形状生成プログラムの等価性の証明の方法論の提案を行ない，有効性確認のために同じ形状を描く2つの異なるプログラムを用いた証明実験を行なった。実験では，2次元図形の等価性証明を応用できる2+1/2次元のレベルのCADシステムを使って証明を行った。始めに，モデルに対する操作に相当する公理系を定義し，プログラムを元にした表明，標準形への変換を行い比較を行った。等価性の最終的な判断は，高さ毎に分類された平面形状の領域が等価と考えられるかどうかで判断した。

今後の課題として，実際の3次元CADシステムを対象とした証明の実現と，計算機システム上での評価システムの実装などが考えられる。また，形状処理以外の設計プロセスや作業プロセスなどを対象とした等価性証明の問題への応用も考えられる。

参 考 文 献

[1] William R. Margren, Formal Specification of Graphic Data Types, ACM Transactions on Programming Languages and systems, vol.4, No.4, October 1982, pages 687－710.

[2] 10303－1 Overview and Fundamental Principles

[3] JIS B 3700-1 (ISO 10303－1)，産業オートメーションシステム及びその統合－製品データの表現及び交換－第1部：概要及び基本原理

[4] HOARE, C.A.R., AND WIRTH, N. An axiomatic definition of the programming language Pascal. *Acta Inf. 2*, 4 (1973), 335－355