グラフィクスとＣＡＤ 28－2
（１９８７・８・２０・２１）

３面図入力を基本にした
３次元形状入力システムの試作

原田　毅士　・　木村　文彦
東京大学工学部

本報告では、形状モデラに３次元形状を簡単に入力できるシステムの実現を目指して、３面図入力を基本した方法である２面図入力方式を提案する。

この方式は、立体の正面図・側面図から３次元モデルを自動的に生成するもので、３面図（正面図・側面図・上面図）入力方式よりユーザが誤った平面図を描く可能性が少なくて済む。

この２面図入力方式に基づいた３次元形状入力システムを、ＳＵＮワークステーション上に作成した。本システムは、形状モデラ部分・平面図入力部分・３次元モデル生成部分の３つから構成され、ユーザは２つの平面図を入力することにより形状モデラに立体を自由に定義することができる。

Solid Model Input System
Based on Orthographic Views

Tsuyoshi HARADA and Fumihiko KIMURA
The Faculty of Engineering, The University of Tokyo
7-3-1, Hongo, Bunkyo, Tokyo, 113 Japan

We propose the solid model input method through two orthographic views, which makes it easy for us to input solid models to geometric modelers (GM).

The input method through two orthographic views is superior to the method through three views, because there are fewer errors in two views.

On SUN work startion, We made solid model input system based on two orthographic views. This system is composed of three parts, geometric modeler part, orthographic views input part, solid model construction part. The users of GM construct solid models more easily than before, using this input system.

1　はじめに

近年、CAD（コンピュータ支援設計）の基礎として形状モデラが利用されているが、そのユーザインターフェースについては貧弱なものが多く、特に3次元形状の入力部分にその傾向がある。すなわち、形状モデラに思いどおりの立体を入力することが難しい場合が多いのである。そこで、私は形状モデラに3次元形状を簡単に入力できるシステムの実現を目指して研究を進め、3面図入力を基本にした3次元形状入力システムを試作した。以下、この報告書では、3次元形状の入力方法である3面図入力とその問題点、2面図入力の特徴、試作した3次元形状入力システムの概要とその構成などについて、順に述べていくことにする。

2　3面図入力とその問題点

形状モデラに立体を入力していく場合、その方法としてはいろいろ考えられるが、主なものとして次に示す2つがあげられる。

(1) 立方体・円柱・球といったプリミティブ（基本立体）の集合演算（和・差・積）を行って、立体を会話的に定義していく方法。
(2) 立体の3面図（正面図・側面図・上面図）を入力して、それら3つの図面から3次元モデルを合成していく方法。

(1) の会話的な3次元形状定義は、現在注目を集めている多くの形状モデラシステムが採用しており、その特徴として自由曲面など定義できる形状の自由度が大きい、ということがあげられる。それに対し、(2) の3面図からの3次元モデル合成においては、平面図に描ける図形が限られているため、入力できる形状の自由度が小さい。しかし、産業的には図面が主流であること、人間は平面上でしか図形を自由に描けないことの2点を考えて、試作したシステムでは立体の入力方法として3面図入力を基本とすることにした。

3面図入力とは、図1に示すように定義する立体の正面図・側面図・上面図を描くことによって、3次元モデルを生成する方法である。この入力方法を用いれば、今まで設計者が図面に描いてきたのとほとんど同じ方法で、立体を自然に形状モデラに入力することができる。

しかし、3面図入力には問題点も多い。入力できる形状の自由度が小さいことは既に述べたが、最も問題になるのは、入力する3つの平面図に「誤り」のないことが前提となっていることである。たとえば、上面図に正面図・側面図と矛盾する部分があった場合、既存の3面図入力システムでは、立体の3次元モデルを合成することが不可能なのである。このような例を図2に示す。3面図入力における「誤り」を検出し修正することは難しい問題であるが、これをうまく解決するために今回試作したシステムでは、2面図入力を立体入力方法として用いることにした。この方法の詳細は次に示すことにする。

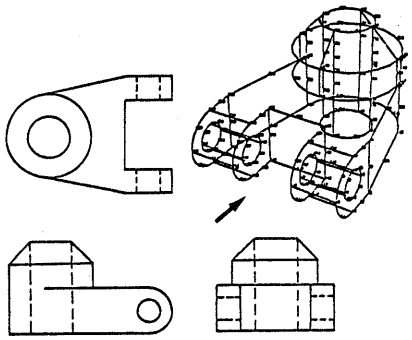

図1　3面図入力

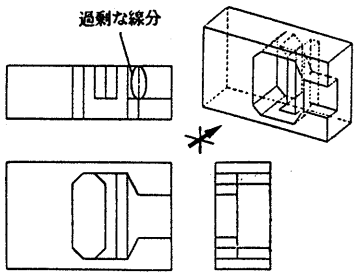

図2　不可能な3次元モデル合成

3　立体入力方法としての２面図入力

前に述べた３面図入力の問題点を解決するために、ここでは２面図入力を提案する。２面図入力とは、図３に示すように正面図と側面図の２つだけから立体を自動的に生成する方法である。そして、主な情報を含む正面図には、「誤り」がないものとシステム側で仮定している。２面図入力による立体入力方法は、次のような長所を持つ。

(1) 立体を生成するために、正面図と側面図の２つだけしか用いないので、ユーザが「誤る」可能性が少ない。
(2) 主な情報を含む正面図を「誤り」のない図面と仮定しているので、もう１つの図面である側面図の「誤り」を検出しやすい。
(3) ２つの平面図より生成された立体から上面図を作りだし、それをユーザが修正するという手法も可能である。このような方法をとることにより、最初から上面図を含む３面図を入力するよりも、「誤り」の混入する可能性が少なくなる。

また、２面図入力の短所としては、以下のような点があげられる。

(1) ３面図入力に比べて立体生成のための情報が少ないために、候補となる立体が数多くできてしまう。
(2) 上から見て非常に複雑な立体は、生成が困難である。すなわち、ユーザは立体のどの部分を正面図にし、どの部分を側面図にするかをあらかじめ考えながら入力する必要がある。

ユーザから与えられる立体の情報は、２面図入力の方が３面図入力に比べて少ないことは明らかである。しかし、少ない情報から計算機がある程度常識的な立体を生成し、ユーザがそれに対して修正を加えていく、という方式を取った方が「誤り」の入る余地が少ないため、立体の入力方法としてより優れている。

4　２面図からの立体生成手法

ここでは、２つの平面図（正面図・側面図）から立体を生成する手法について詳しく述べていくことにする。最初に基本的な用語について説明し、そのあとで図を用いながら、実際に立体を生成する手法を解説する。

4－1　基本的な用語

この項では、基本的な用語について解説する。ただし、２つの平面図のうち、正面図は３次元空間上のＸＹ平面、側面図はＺＸ平面上に描かれるものとする。

(1) C-Vertex (Candidate Vertex)
２つの平面図の中にある点から生成される、立体の頂点の候補。（図４）　C-Vertex の中から立体の本当の頂点が選び出される。
(2) C-Edge (Candidate Edge)
２つの平面図の中にある線分から生成される、立体の稜線の候補。（図４）　C-Edge の中から立体の本当の稜線が選び出される。
(3) V-C-Edge (Vertical Candidate Edge)
C-Edge のうち、ＹＺ平面に水平な面上にのっていないもの。（図４）
(4) H-C-Edge (Horizontal Candidate Edge)
C-Edge のうち、ＹＺ平面に水平な面上にのっているもの。（図４）

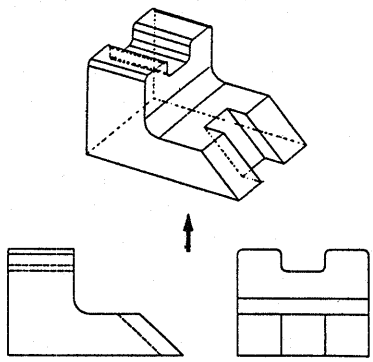

図３　２面図入力

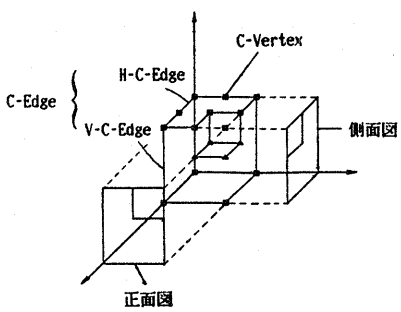

図４　C-Vertex と C-Edge

4－2　立体生成手法

正面図・側面図の２つから、立体の３次元モデルを生成するときに、６つの段階がある。以下、図５に示した立体の生成を例にとりながら、それぞれの段階について説明していく。

(1) C-Vertex の生成
最初に、２つの平面図から可能な C-Vertex をすべて生成する。（図６）　この中には、余分な頂点も含まれている。

(2) V-C-Edge の生成
正面図、側面図の線分から、可能な V-C-Edge をすべて生成する。（図７）

(3) C-Vertex の分類
C-Vertex を同じＹ座標を持つグループに分類する。（図８）　そのグループ内で、H-C-Edge を生成することになる。

(4) H-C-Edge の生成と、C-Vertex, V-C-Edge の削除
(3) で分類した、同じＹ座標を持つ C-Vertex のグループから、H-C-Edge を生成し、余分な C-Vertex と V-C-Edge を削除する。（図９）　H-C-Edge を生成する方法として、図１０に示すように、決められたパターンをデータベースに保存し、それに基づいて H-C-Edge を生成するという手法を取っている。

(5) 生成された C-Vertex, C-Edge のチェック
正しい立体は、１つの頂点に必ず３つ以上の稜線が集まっている。(4) までで生成された C-Vertex,C-Edge をこの法則に基づいてチェックし、正しい立体になっていなければ、(4) にもどって、もう１度 H-C-Edge を生成し直す。

(6) 立体の３次元モデルの生成
(5) まで終わったところで、立体の頂点と稜線は、でき上がっているはずである。あとは、面の情報を作り出せば、３次元モデルを生成できる。これは、Wire-Frame の情報から、Solid Model を生成する手法であり、参考文献 [2] [3] などに詳しい。（図１１）

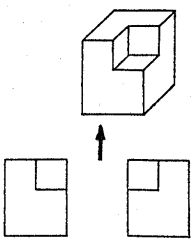

図５　２面図入力の例

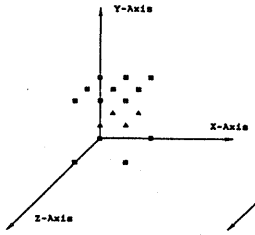

図６　C-Vertex の生成

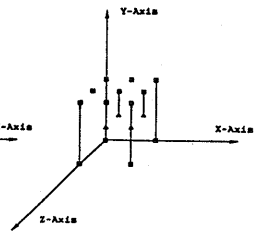

図７　V-C-Edge の生成

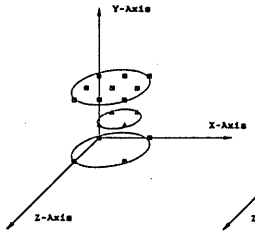

図８　C-Vertex の分類

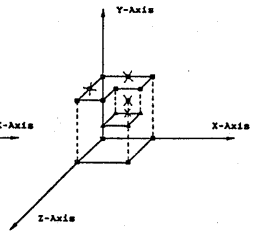

図９　H-C-Edge の生成と C-Vertex, V-C-Edge の削除

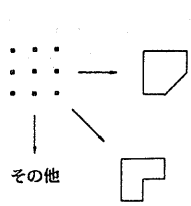

図１０　H-C-Edge のパターン

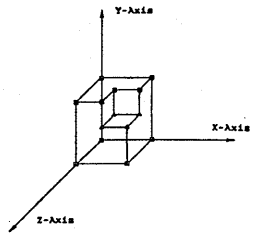

図１１　３次元モデルの合成

5　試作した3次元形状入力システムの構成

4で述べた手法を用いて、SUNワークステーション上に3次元形状入力システムを試作した。このシステムは、図12に示すように、主に次の3つの部分から構成される。

（1）3次元形状入力システムの基本となる形状モデラ。
（2）2つの平面図（正面図・側面図）の入力部分。
（3）2つの平面図から3次元モデルを生成する部分。

以下に、3次元形状入力システムを構成するこれら3つの部分について、その詳細を説明していくことにする。

5－1　本システムの基本となる形状モデラ

3次元形状を入力するシステムを考える場合、基本となる形状モデラが必要である。しかし、既存の形状モデラを利用した場合、そのデータ構造に対する理解不足や、形状モデラ自体の変更が不可能であるといった大きな問題が生ずる。そこで、本システムの基礎となる形状モデラを自分で作成した。

本システムで用いる形状モデラの主な特徴を、次に示す。

(1) モデラは多面体のみを扱う。すなわち、稜線形状は直線、面形状は平面しか扱わない。（これは、近い将来拡張する予定である。）
(2) 3次元形状のデータ構造は、HALF-EDGEデータ構造（WINGED-EDGEデータ構造をもっと扱いやすくしたもの）とする。（参考文献 [4] 参照）
(3) 基本的な形状操作は、オイラー・オペレータによるものとし、行った形状操作の逆操作も可能にする。

図13、図14、図15に、SUNワークステーション上で作成した形状モデラの画面を示す。図13は、2次元形状を入力したときの画面であり、図14は、それを立ち上げて3次元の立体を作り、穴を開けたとき画面である。また、図15は別の立体を作成したものである。このように本モデラは、多面体モデラとしては一応の基本機能を備えている。すなわち、2次元平面上で図面を作成し、それを立ち上げ、また穴を開けたりすることが、この形状モデラを用いて可能である。

図16は、図15で示した立体をファイルに書き出したときの出力データ形式（一部）である。このように、3次元立体はオイラー・オペレータの列として保存されることになるので、他の形状モデラとのインターフェースも比較的つくりやすい。

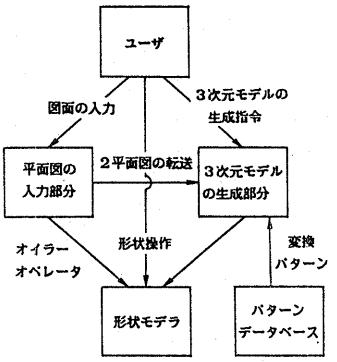

図12　3次元形状入力システムの構成図

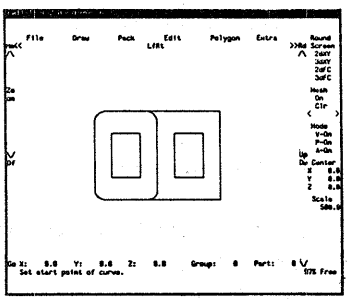

図13　形状モデラの画面（1）

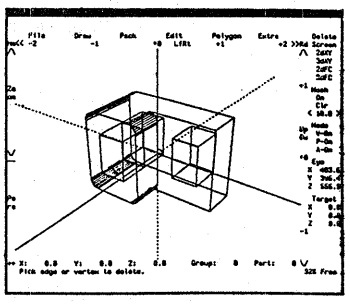

図14　形状モデラの画面（2）

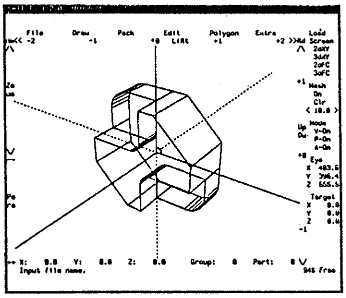

図15　形状モデラの画面（3）

mvfs	35	48		100	30	-50
mev	35	48	47	30	100	-50
mev	35	48	74	100	30	50
mev	35	74	75	30	100	50
mef	75	47	35	33		
mev	35	47	46	0	100	-50
mev	35	75	76	0	100	50
mev	35	76	9	0	100	0
mef	9	46	35	5		
mev	35	46	71	-60	100	-50
mev	35	9	14	-60	100	0
mef	14	71	35	30		
mev	35	71	70	-67.8	99.2	-50
mev	35	14	21	-67.8	99.2	0
mef	21	70	35	29		

図16　出力ファイルの形式

5－2　2つの平面図の入力部分

基本となる形状モデラの上に、2つの平面図（正面図・側面図）の入力部分を作成する。

2面図入力を考えた場合、ユーザが平面上で図形を入力し、しかる後に、立体の3次元モデルに変換していくということになる。その2次元形状の入力部分は、次のような特徴を持っている。

(1) 形状の入力方法としては、SUNワークステーションの持つマウスと、それを補助する意味でのキーボードの2つをサポートする。
(2) 3次元空間の任意の平面上に図形を描くことができるようにする。その中で特に重要なのがXY平面、YZ平面、ZX平面上での入力である。

本システムのこの部分に、パラメトリックデザインの手法を取り入れれば、形状入力のシステムとしては、非常に強力なものになると思われるが、それは将来への課題であろう。

5－3　2つの平面図から3次元形状を生成する部分

この部分が、試作した3次元形状入力システムの中核である。2つの平面図から3次元モデルを生成する手法は4に述べたので、ここでは、特徴的な部分をあげることにする。

(1) 正面図、側面図の2つの平面図から、3次元形状を生成することを基本にする。
(2) システム側では、正面図を常に誤りのない図面と仮定する。側面図に正面図と矛盾する場所があった場合、システムはその旨ユーザに告げる。
(3) もし正面図と側面図から3次元モデルが合成されたならば、システムはそのモデルから上面図を自動的に作成する。ユーザが、その上面図を修正していけば、より完全な3次元モデルを合成できる。
(4) 正面図・側面図から生成できる3次元形状のパターンをデータベースに保存できる。将来は、このパターンをユーザが自由に編集、修正できるようにする。

人間側で、いかに入力の労力を少なくするかということは、システムでいかに多くの仕事をするかということにかかっている。3次元形状を入力する場合、人間側の労力が多くなりがちなので、システム側である程度その肩代りをするということが、本システムの大きな目的である。

6　実行例

図17に試作したシステムの全画面を示す。左側のウインドウが、形状モデラ部分と平面図入力部分であり、右側が2面図から3次元モデルを合成する部分である。形状モデラと平面図入力部分はC言語で、3次元モデル合成部分はLISP言語で書かれており、それらの間のインターフェースはファイルによるデータ転送となっている。

図18から図21までに、実際に2つの平面図を入力して3次元モデルができあがるまでの流れを示す。これからわかるように、2面図入力はユーザにとってかなり楽に立体を入力できる方法ではないか、と思われる。生成された3次元モデルから上面図を作り出し、それを修正することも可能であるため、ユーザが誤った線分を描いたりする可能性も少ない。

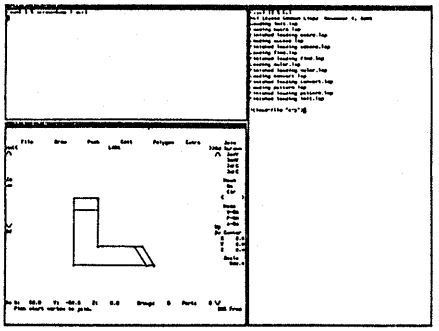

図１７　本システムの全画面

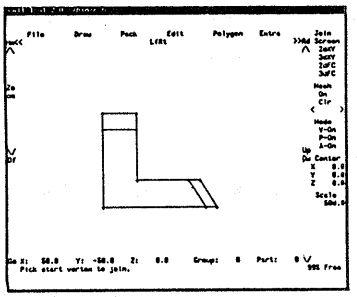

図１８　形状モデラに入力した正面図

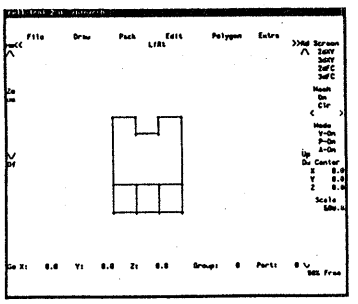

図１９　形状モデラに入力した側面図

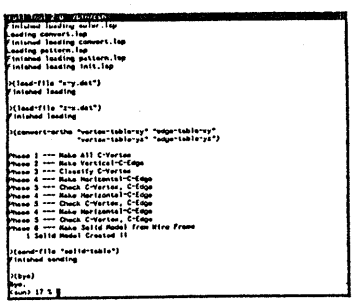

図２０　LISPシステムによる
３次元モデルの生成

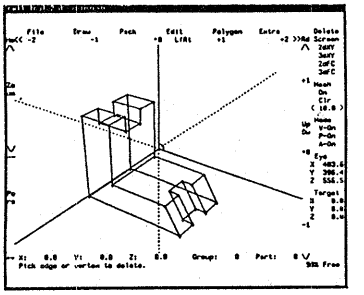

図２１　生成された立体の
３次元モデルの表示

７　結論、展望

この報告書では、３面図入力とその問題点、２面図入力と立体生成手法、試作した３次元形状入力システムの概要について述べてきた。その中でも、特に２面図入力が立体入力方法として優れていることを示した。

しかし、まだ多面体しか対応できていないこと、生成できる立体のバターンが少ないことなど問題点も多い。形状モデラで円弧とそれから発生する面を扱えるようにすること、生成できる立体のバターンをユーザが自由に編集できるようにすること、がこれからの課題である。

［参考文献］

[1] Hiroshi Sakurai and David C.Gossard：
"SOLID MODEL INPUT THROUGH ORTHOGRAPHIC VIEWS"
Computer Graphics,July 1983

[2] Patrick M. Hanrahan：
"Creating Volume Models from Edge-Vertex Graphs"
Computer Graphics,July 1982

[3] S. Mark Courter and John A. Brewer：
"Automated Conversion of Curvilinear Wire-Frame Models to Surface Boundary Models; A Topological Approach" ACM SIGGRAPH,Volume 20,Number 4,1986

[4] Martti Mantyla：”An Introduction to Solid Modeling”;1984

討論

２．３面図入力を基本にした３次元形状入力システムの試作　　原田（東大）

浜川：２面（正面図と側面図）を入力してから行う上面の修正は、人間がやるのでしょうか。

原田：上面はシステムが自動的に２面から生成し、それを人間が修正します。３面とも人間が入力すると誤りが多くなるので、このようにすれば効率よく入力することができます。

浜川：３面図入力で問題となるのは図面間の線の対応づけですが、それはこのシステムではどうなっていますか。つまり、線が少しずれている場合にも正しく対応づけられるのでしょうか。

原田：現在は誤差は考慮していません。これから検討して行きます。

浜川：側面図の誤りはどの程度検出できるのでしょうか。

原田：現在は２、３本の線の誤りまで検出できます。

川合：正面図と側面図から生成できる３次元形状のパターンを見つけるのは簡単にできるのですか。

原田：実はその条件が難しいのですが、１つの頂点に３本以上の線が集まらなければならないという条件で大部分が除けます。さらに、平面性の条件を加えることが考えられますが、このシステムではそこまでは入れていません。

川合：このパターンは人手で作るのですか。

原田：そうです。１０点×１０点以上の複雑なものはまだ対処していません。
また、パターンを検索する順番をユーザが変えられるようにすると効率よく望むべき立体が得られるので、そのような拡張を考えています。

福井：桜井さんの方法（参考文献［１］）とどこが違うのですか。

原田：桜井さんの方法は３面とも入力するので、そこが違います。また、本システムは位相的な情報だけから面を張っています。

守屋：形状に幾何学的な情報以外の情報、例えば、物の情報などを持たせてはどうでしょうか。

原田：現在は考えていません。

近藤（東芝）：どの線が隠線かを教えてやれば、立体の構成に役に立つのではないかと思います。