グラフィクスとCAD 16－3
（１９８５．３．１２）

自由曲面立体の対話的設計環境

An Interactive Design Environment for Free-Form Surface Solids

千代倉 弘明　　植田 健治
Hiroaki CHIYOKURA　Kenji UEDA

(株) リコー　ソフトウェア研究所
Software Research Center, Ricoh Co., Ltd.

Two facilities are proposed to assist designers in an interactive modeling of free-form surface solids. The first facility is a design environment that naturally lends itself to an interactive specification of UNDO and REDO operations. A history substitution mechanism like the C shell, a command interpreter of Unix operating system, is also shown. The second facility is a rounding operation for interactively modifying solid models. These facilities have been implemented in our solid modeling system DESIGN BASE.

1. はじめに

多くの工業製品の形状には様々な自由曲面が含まれており，それらは機能的，美観的に重要な役割を演じている．このような自由曲面を設計するシステムは，従来数多く開発されてきたが，これらのシステムの多くは内部表現として面モデルを用いている．しかし，応用面からみると形状が立体モデルとして表現されていることのほうが好ましく，曲面を含んだ立体を対話的に設計できるようなシステムが望まれる．
そこで，著者らは，自由曲面立体の対話的設計環境において，設計者の作業を支援する二つの新しい機能を提案する．一つは，逆操作(UNDO)，再操作(REDO)を行えるようにするための新しい立体生成過程の表現である．もう一つは，多面体の稜線の頂点を丸め，容易に自由曲面立体を生成するための丸め変形操作である．このような機能は，UNIX 上に Cを用いて開発中である DESIGN BASE の中に実現された．

いくつかのプログラム開発システム[1][5]では，ＵＮＤＯのようにシステムの過去の状態に戻るための機能が備わっている．また，一般的な，コマンドのＵＮＤＯやＲＥＤＯの枠組みの研究もなされている[7]．立体モデル生成システムにおいてもこのような機能は極めて重要である．そこで，我々は立体生成過程の新しい表現法を導入しＵＮＤＯ，ＲＥＤＯが自由に行えるようにした．

すでに筆者が提案した丸め変形操作[2]では，多面体を丸める際にいくつかの制限があった．また，丸めの半径を指定することもいくぶん難しかった．しかし，新しい丸め変形操作においては，これらの問題は解決されている．

2. 立体の生成過程の表現方法

2.1. 変形操作の逆操作(UNDO)と再操作(REDO)

我々のシステムにおける立体形状の境界表現はいつも，基本変形操作[3]によって生成・変更されている．したがって，局所変形操作や集合演算は，この基本変形操作によりその境界表現を変化させる．基本変形操作はすべて，対応する逆操作を持っており，立体変形において用いられた基本変形操作列は立体生成過程の表現の中に蓄えられるので，すべての立体変形操作の逆操作が可能になる．

立体生成の過程は木構造で表現され，その枝の部分に変形操作に対応する基本変形操作列が蓄えられる，この木構造をたどることで，自然に変形操作のＵＮＤＯや，ＲＥＤＯといった設計過程における試行錯誤的な動作を支援することができる．すなわちＵＮＤＯとは，木構造を根の方向にたどることであり，ＲＥＤＯは葉の方向にたどることと解釈すればよい．もちろん，普通の変形操作の実行は，枝を伸ばし新しい形状に対応する葉を作ることである．

2.2. 単一立体の生成過程

立体を生成する場合の変形操作として，単一立体内での変形操作と，二つの立体に対する操作の二つに分けて考える．ここではまず，単一立体を生成する時の生成過程がどのようにシステム内に表わされるかを示す．

設計者がある立体を生成するときには，変形コマンドを次々にシステムに入力しながらおこなってゆくが，それと同時に，システム内では木構造がつくられ変形操作が書えられてゆく．立体の初期化コマンドで，木の根の部分が作られ，変形操作の実行とともに枝が伸び新しい葉ができあがってゆく．このとき，枝にはその変形操作に対応する基本変形操作列が書えられる．また，新しい葉には自動的に識別名が付けられる．この名前はあとで，その形状を指し示すために使用される．そして，その立体の状態が新しい形状を示すようになる．このような構造にしておくと，ある状態から直前の形状に戻るには，枝を逆にたどりながら，その枝に書えられた変形操作の逆操作を行えばよいことになる．逆に，ある形状の次の状態に至るには，木構造を葉の方向にたどりながら枝の中の基本変形操作列をもう一度実行すればよい．このＵＮＤＯとＲＥＤＯを組み合わせればそれまでに存在した形状であれば，そのどれにでも到達できる．

設計者はこの木構造を眺めながら，ＵＮＤＯとＲＥＤＯを組み合わせて，それまでにあった任意の形状を取戻すことができる．それに加えて，識別名を指定すればシステムが自動的にその状態まで変形操作を実行してくれるようにもなっている．この表現は，設計者が自由に形状にアクセスできるとともに，誤った操作の実行からの回復を容易にする結果をももたらす．

図１は，立体Ａの生成過程と，その表現木である．

１） まず生成過程の表現木の根の部分が作られる．これは空の立体を意味する．そのあと最初の形状を示す節（葉）が作られ，Ａ1,1 という識別名が付与される．これは，形状が立体Ａの第一期の一番目であることを示している．

２） 立体Ａの面Ｆ１に稜線ができ，Ｆ１がＦ１とＦ２に分割される．これでできた形状には，Ａ1,2 という識別名が付けられる．その後，Ｆ１が持ち上がり，最後に稜線Ｅ１が丸められる．

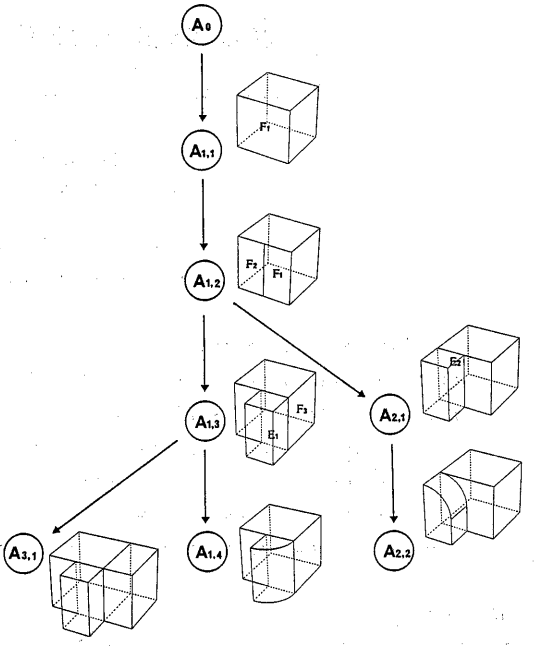


図1　単一立体の変形操作

３） ＵＮＤＯが二度実行され形状 Ａ1,2 が再現されたあと，今度は，Ｆ２が持ち上げられる．このようなＵＮＤＯの後の新たなコマンドの実行では，期の番号が増やされ新しい形状には，Ａ2,1 という識別名が付く． その後 Ｅ２が丸められて出来た形状には Ａ2,2 という識別名が与えられる．

４） ＵＮＤＯを二度，ＲＥＤＯを一度実行して Ａ1,3 を再現して，面 Ｆ３を持ち上げた形状は Ａ3,1 で識別されることとなる．

このように，システム内で立体の生成過程が保存されることで，それ以前に作られた形状の境界表現がすぐに再生できるようになる．このＵＮＤＯまたはＲＥＤＯのための計算時間は，一般のコマンドのような基本変形操作以外の計算や，入力データのチェック等の手続きを必要としないので，極めて小量である．このことは，対話的な設計環境に，立体生成過程のここで述べたような表現を導入することの妥当性を示している．

2.3. 複合立体の生成過程

単一立体の変形操作のあと，立体集合演算[6]や接合操作のような複数の立体に対する操作でも木構造が作られる．図２は，立体Ａと立体Ｂから立体Ｃが生成された場合を示している．ここで，⊕は，複数立体への操作を示していて，上方への枝は，立体Ｃ1,1 と接続し，下方への枝は，立体Ａ2,2 と立体Ｂ1,3 に接続している．この操作での基本変形操作列は，⊕に接続している枝に蓄えられている．

ここでも，設計者はこのようなシステム内での変化は気にせずに，新しい立体Ｃを生成したということを記憶しておけばよい．このために，複合立体を作るような操作を行うときには，新しくできる立体には新しい名前を付けることを，設計者は要求される．このことで，設計者はそれまでに生成した立体の識別が確かなものとなる．

2.4. 入力されたコマンドの再利用

この立体モデル作成システムは，ＵＮＩＸオペレーティング・システム上で開発されているが，設計作業の効率を上げるものとして入力コマンドの再利用のためにＵＮＩＸのコマンド・インタプリタ C shell のヒストリ置換に準じたヒストリ置換機能を実現している．これは，それまでに入力されたコマンドを，その番号とともに蓄積しておいて，新しい入力文字列中に記憶された文字列の一部分を指すような特別な文字列が現われたならば，その文字列で置き換えてコマンド解析部に渡す機能である．その特別な文字列は C shell と同じように，文字 '!' で始まるもので，そのあとに続く文字列で過去の文字列を指したり，それを更に修正したものを表わすことができる．例えば，以下のような指定方法がある．

!12	12番目のコマンド
!-4	四つ前のコマンド
!!	直前のコマンド ('!-1' と同じ)
!12:1-3	12番目のコマンドの一番目から三番目までの引数
!-4:2-4:s/1/r/	四つ前のコマンドの二番目から四番目までの引数部分の '1'を'r'で置き換えたもの

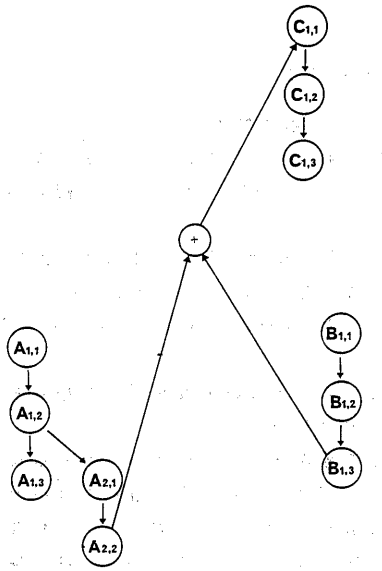


図2　複合立体の変形操作

設計者は，このような文字列を入力コマンドの中に挿入することで，過去に入力した文字列を再び利用できる．この機能により類似のコマンドの入力や，誤ったコマンドを入力した場合等において，過去に入力した文字列の一部を再使用することができるので，設計者はキーボードからの入力の手間を減らすことができる．ＵＮＤＯ，ＲＥＤＯコマンドは，一般のコマンドの実行を行うのではなく，過去の変形操作を利用して形状を再生するためのものであるので，コマンドの繰り返し実行や，それまでの入力コマンドと同じコマンドの実行にはこのヒストリ置換を利用する．立体生成過程の保存とヒストリ置換を組み合わせて，設計者は効率的に立体を生成することができる．また，蓄積されたコマンドはファイルに出力して保存することができ，そのようなファイルを読み込んでコマンドを実行することもできるようになっている．エディタでコマンド列を作成しておいて，それを実行することももちろんできる．このようにして，オペレーティング・システムとＣＡＤシステムとのコマンド体系の整合がはかられていると，設計者のシステムの習得にかかる負荷の軽減にもつながる．

3. 丸め変形操作

3.1. 旧丸め変形操作との比較

旧丸め変形操作[2]を用いて自由曲面立体を生成する時には，以下に示すような問題があった．

(1) 図3(a)に示されているような凹な面に接している頂点（inside corner）や，穴に接している頂点（hole corner）を丸めることが，容易ではなかった．

(2) 丸められる稜線の曲率半径を指定することが容易ではなかった．

新丸め変形操作ではこのような問題は解決されている．図3(b)は，図3(a)の立体から，新丸め変形操作を用いて生成された自由曲面立体を示している．また，この操作も他の変形操作と同様に基本変形操作によって実現されているので逆操作が可能である．

3.2. 丸めの指定

丸め変形操作では，設計者は立体の各稜線に，その稜線を丸めるか，そのままにするかによって，それぞれ0または1のフラグをつける．このとき，丸めの半径を指定することもできる．図4のように，稜線(E1)の両端点(V1, V2)に異なる半径を指定できる．もし，何も指定がなければ半径のデフォルト値が指定されたものとみなされる．頂点を丸める時には，その頂点に接している稜線のフラグを0にする．図5(a)は，頂点V1が接する稜線E1，E2，E3に，丸めフラグ0が与えられ，E1，E2，E3に異なる丸め半径が指定されている例を示している．この立体に丸め変形操作を施すと，いくつかの直線の稜線は，曲線に置きかわる．図5(b)では，E1のかわりに，曲線C1,1とC1,2が，E2とE3のかわりに，C2,1，C2,2，C3,1，C3,2が生成されている．ここで，C1,1，C2,1，C3,1の半径は，設計者が指定した値であるのに対して，C1,2，C2,2，C3,2の半径は，システムが決定した平均的な値である．この時，一つの稜線の半径を，設計者が指定することが可能である．その場合には，もとのある稜線が優先稜線であると指定する．もし，稜線E1が優先稜線であるとすると，図5(c)のような曲面立体が生成される．

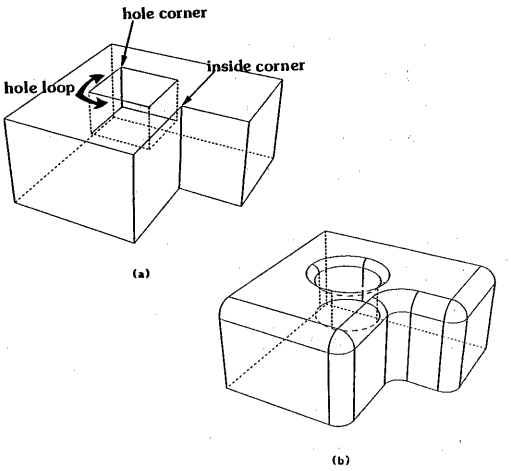


図3 hole corner と inside corner

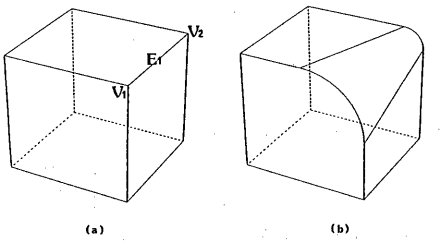


図4 異なる半径での丸め変形操作

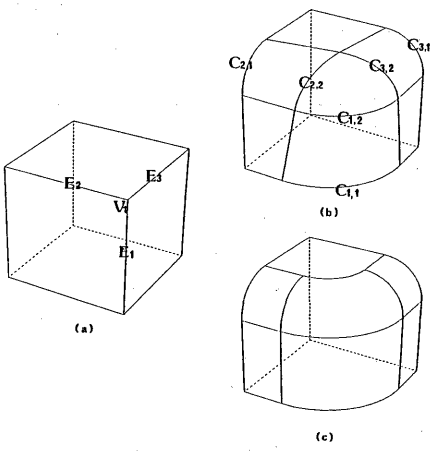


図5 頂点の丸め変形操作

3.3. 丸め変形操作のアルゴリズム

3.3.1. アルゴリズムの概要

丸め変形操作は，次のような段階を経て行われる：

［第１段階］稜線上の頂点の生成

図６(a) の，0 が指定された稜線が丸められる場合には，図６(b) のように，立体の各稜線上に，一または二個の頂点が生成される．これらの頂点によって，稜線はいくつかの部分に分割され，各部分は，もとの稜線に与えられた丸め情報を受け継ぐ．

［第２段階］面上の稜線の生成

図６(c) のように，新しくできた頂点間に稜線が生成され，もとの面を分割する．

［第３段階］曲線稜線の生成

図(d) のように，丸め情報に従って不要な稜線が取り除かれたあと，曲線が生成される．

［第４段階］曲面の内挿

最後に，内挿アルゴリズム［３］によって，曲面が生成される．以下の部分では，それぞれの段階について説明する．

3.3.2. 稜線上への頂点の生成

図７は，丸めたい稜線 E0 と，それに接続している稜線 E1，E2，E3，E4 が示されている．E0 の両端点 V1 と V2 には，それぞれ半径 r1 と r2 が設計者によって指定されているとする．稜線 E1，E2，E3，E4 に対する分割点 P1，P2，P3，P4 は，第３段階で生成される曲線 C1 と C2 の半径が r1，r2 となるように計算される．このとき，もし稜線 E3 にも丸めフラグ 0 が立っていたなら頂点 P4 は，二回計算されることとなる．このような場合，P4 は，二点間の中点となる．このような手続きが，丸めたい稜線すべてに対して行なわれる．この手続きによって稜線に分割点が求まり，最後に基本変形操作を用いて頂点が作られる．

3.3.3. 面上での稜線の生成

図８(a) には，元の頂点 Vi (i=1,…,6) と，第１段階で新しく生成された頂点 Ni (i=1,…,7) を含むような面 F1 が示されている．ここで，もとの頂点は V1，V2 のような凸頂点と，V3 のような凹頂点とに分類できる．凹頂点 V3 からは，稜線 E1 と 頂点 V1 が新たに生成

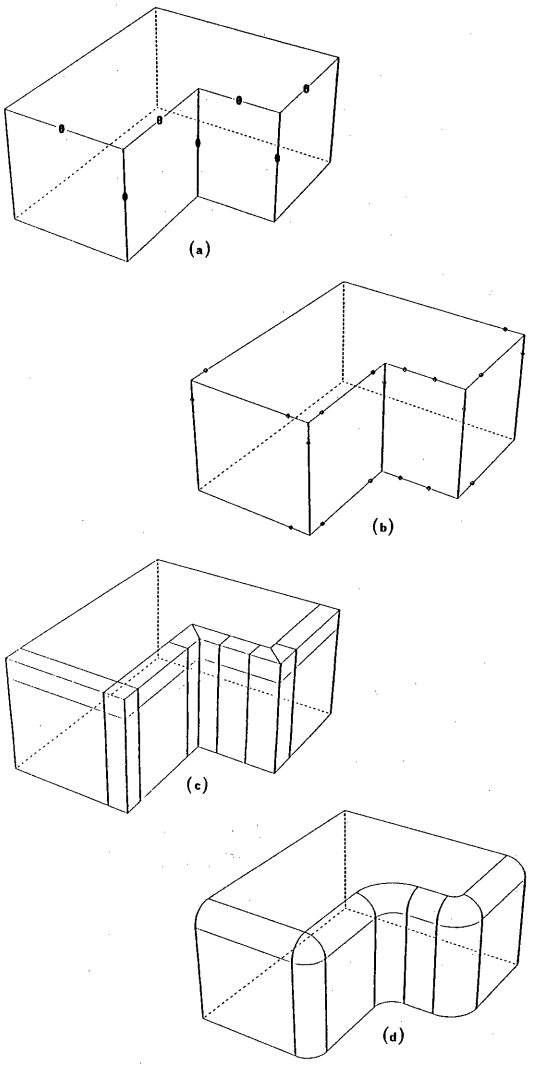


図6　丸め変形操作の例

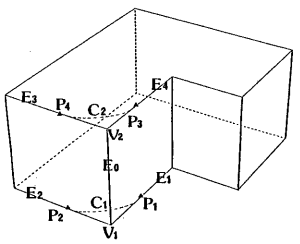


図7　稜線の丸め変形操作

される(図8(b)). もし，頂点 V4 に接続している稜線 E4 が優先稜線であれば，3.2. で述べたように稜線 E5 と頂点 V8 が新たに生成される．次に，図8(c) にあるように，新しい頂点間に稜線が生成される．これが終わった後でも，N4 や N5 のような 頂点が稜線を形成せずに残っているので，図8(d) に示すように，これらの頂点と新たにできた稜線との間に稜線が作られる．このような手続きは，すべての面と，新しい頂点を持つ穴ループ（図3）に対して行なわれる．

3.3.4. 曲線稜線の生成

図9は，立体上の頂点 V1 を示していて，それは，稜線 E1，E2，E3，E4 と接続している．これらの稜線はまた，それぞれN1，N2，N3，N4 というもう一方の頂点を持っている．このとき，次のような手続で V1 に接続している稜線が，削除され，新たに曲線の稜線が生成される．

[手続1] もし，1 とラベル付けされた稜線が一つ以下で，残りは0とラベル付けされているなら，図10(a) のように，頂点 V1 とすべての稜線を削除する．

[手続2] もし，二つの稜線 E1，E3 だけが 1 とラベル付けされていれば，図10(b) のように，頂点 V1 とすべての稜線を取り除いたあと，頂点 N1 と N3 の間にこれらの頂点でそれぞれ稜線 N1V1，N3V1 と接するような曲線の稜線を生成する．

[手続3] もし，0 とラベル付けされた稜線がただ一つであれば，図10(c) のように，その稜線を削除したあと，頂点 N1 と N3 の間に曲線の稜線が生成される．

[手続4] もし，すべての稜線のラベルが1ならば，何もしない．

これらの手続きが，もとあった頂点すべてに適用されたならば，図11(b) のような立体が図11(a) から得られる．この場合，頂点 V2，V3，V4 には手続1が，頂点 V1，V5 には手続2が，それぞれ適用される．図11(b) で，■が付いている頂点は，第1段階で生成されたものである．最後に，これらの頂点に接続する稜線は取り除かれ（図11(b) で ×のついたもの），曲線が生成されている．この結果，図11(c) のような曲面立体が得られる．

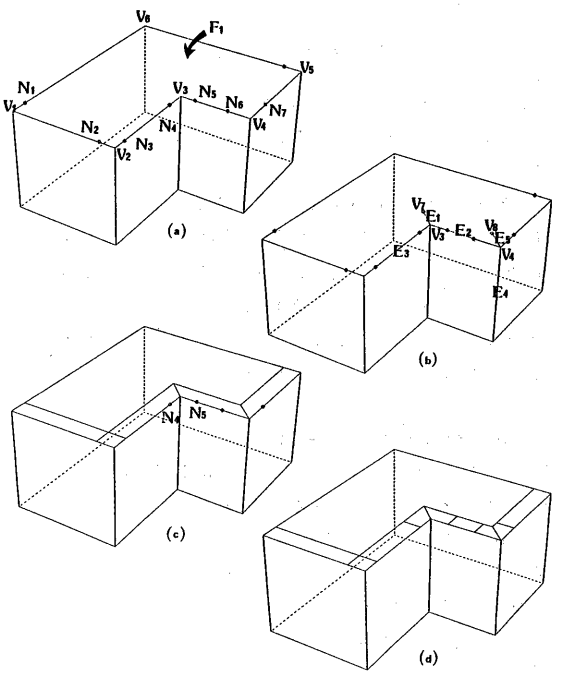


図8 面内での稜線の生成

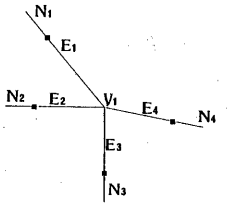


図9 頂点の丸め

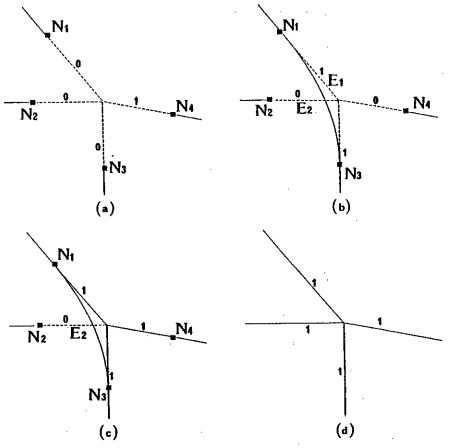


図10 種々の頂点の丸め

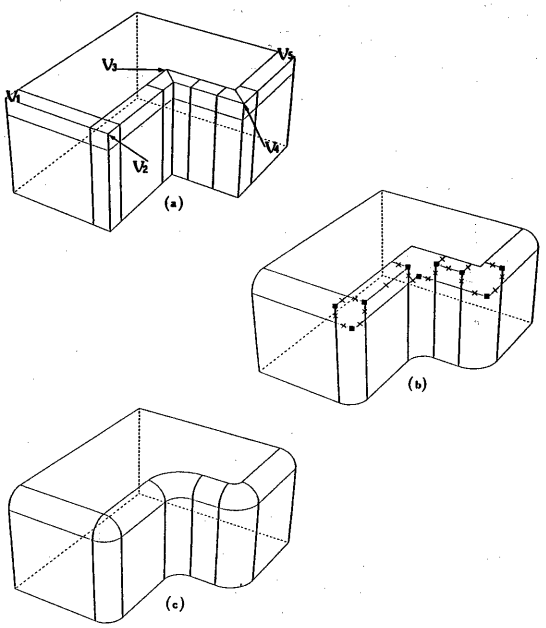


図11　第一段階で生成される頂点

図12は，我々のシステムで生成した機械部品であり，(a) は，立体の原形，(b) は，丸め変形を行ったあとの形状，(c) は，光線追跡法を用いて出力した陰影画である．図13は，同様のオフィス・コンピュータである．

4. おわりに

いくつかの，自由曲面立体生成において設計の助けとなる機能を説明してきた．これらの機能によって自由曲面立体の対話的設計環境が改善されるが，それぞれに未だ改善すべきことがらが残されている．特に，新しい立体生成過程の木構造による表現の研究は緒についたばかりである．ＣＡＤにおける，設計者とシステムの対話的環境はまだまだ改善される必要があり，このような状況において，我々が採用した，立体生成過程の表現の潜在能力は大きいと考える．

謝辞

本研究を行う上で多くの有益な助言をいただいた東京大学工学部木村文彦助教授，ならびに國井秀子所長に感謝いたします．

参考文献

[1] J.E.Archer, Jr., R.Conway and F.B.Schneider, "User Recovery and Reversal in Interactive System," ACM Trans. Programming Languages and Systems, Vol.6, No.1, January 1984, pp.1-19.

[2] H.Chiyokura amd F.Kimura, "Design of Solids with Free-Form Surfaces, "Computer Graphics (Proc. SIGGRAPH '83),Vol.17, No.3, 1983, pp289-298.

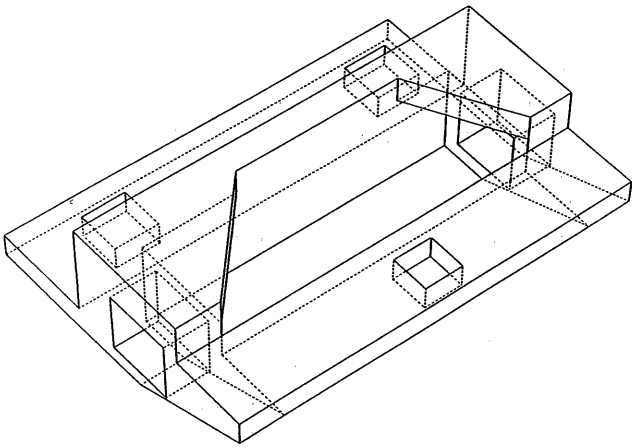
[3] H.Chiyokura amd F.Kimura, "A Representation of Solid Design Process using Basic Operations," Computer Graphics Tokyo '84 Proc., T4-6, April 1984.

[4] H.Chiyokura and F.Kimura, "A New Surface Interpolation Method for Irregular Curve Models," Computer Graphics Forum (J. of EURO-GRAPHICS), Vol.3, No.3, 1984, pp.209-218.

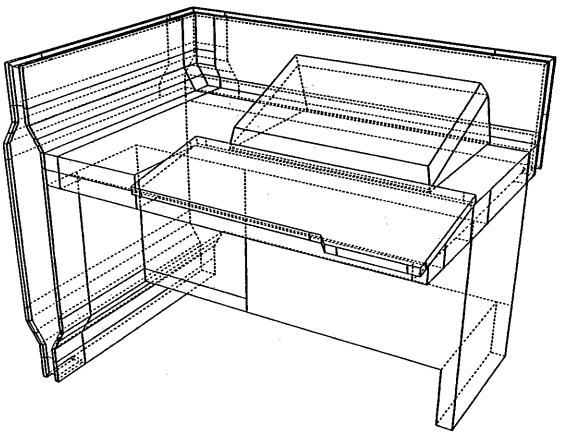
[5] W.Teitelman, Interlisp Reference Manual, Xerox Palo Alto Reaearch Center, revised 1978.

[6] 鳥谷 浩志，佐藤 敏明，千代倉 弘明，"逆操作を持った立体集合演算，" 情報処理学会第16回 グラフィックスとＣＡＤ研究会，16-4，1985.

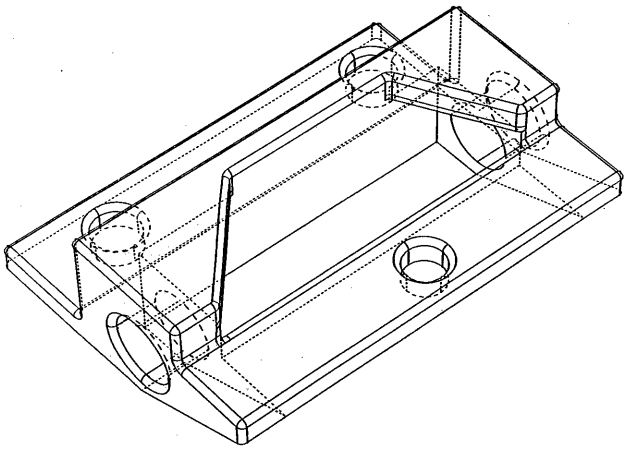
[7] J.S.Vittor, "US&R: A New Framework for Redoing," IEEE Software, Vol.1, No.4, October 1984, pp.39-52.



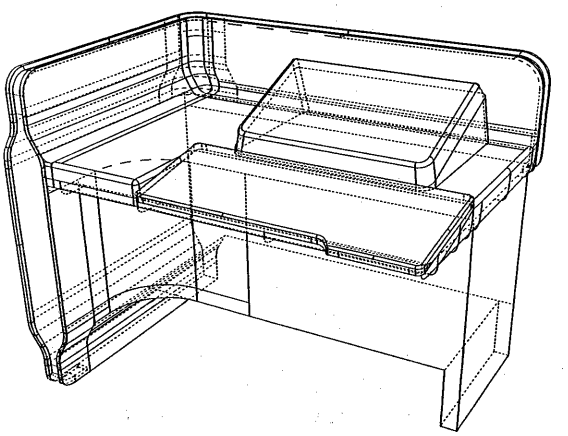
(a)



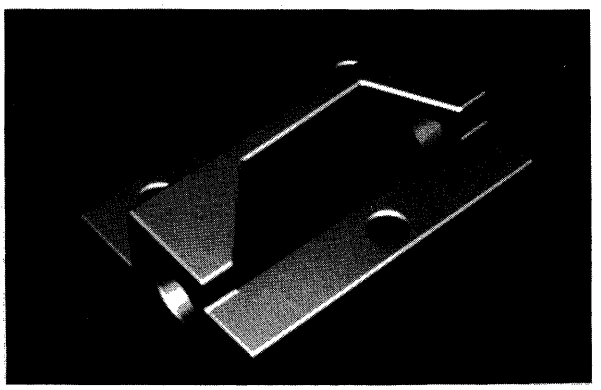
(a)



(b)

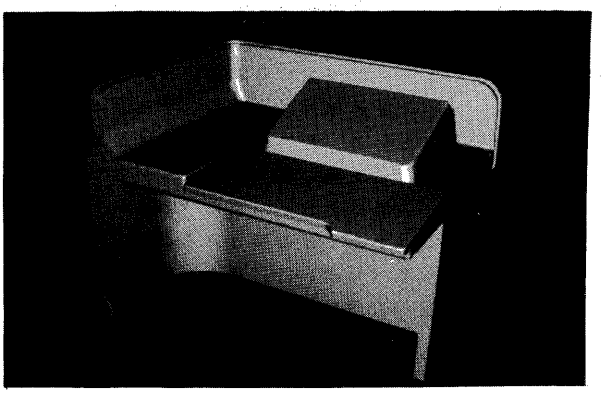


(b)



(c)

図12　機械部品



(c)

図13　オフィス・コンピュータ