手書きスケッチによるモデリングシステム Teddy

五十嵐 健夫[†], 松岡 聡[‡], 田中 英彦[†]

[†]東京大学 情報工学専攻,
東京都文京区本郷 7-3-1
Tel: 03-3812-2111 (ex.7413), Fax 03-5800-6922
Email: takeo, tanaka@mtl.t.u-tokyo.ac.jp

[‡]東京工業大学 数理・計算科学専攻
東京都目黒区大岡山 2-12-1
Tel/Fax: 03-5734-3876
Email: matsu@is.titech.ac.jp

概要

ディスプレイ付きタブレットやマウス等を利用し、平面上に手書き自由曲線を描いていくことにより 3 次元モデルを対話的に構成する手法について紹介する。本手法により、コンピュータグラフィクスの知識のない初心者でも、紙と鉛筆をつかってスケッチするのと変わらない手間で自由に 3 次元モデルを生成することが可能となる。既存のスケッチによるモデリングシステムが、主に直方体のように平面で囲まれた人工的なオブジェクトの生成を目的としていたのに対し、本手法は動物や植物といった曲面を主体とするオブジェクトの生成を目的としている点で革新的である。具体的には、2 次元の手書き閉曲線を縦方向に膨らませることで新たにモデルを生成する手法、手書き自由曲線により物体を切断したり隆起させたりする手法等について説明する。手書き風リアルタイムレンダリングを利用したプロトタイプシステムが JAVA アプレットとして実装されており、WWW 上に公開されている。

1. はじめに

3 次元モデルの作成は依然として困難な作業である。通常の３次元モデリングソフトウェアを使用するためには、十分な時間をかけて操作を習得し、さらに複雑な編集コマンドを使用したり多数の制御点の位置を調整しながらモデルを生成していかなければならないなど、特に初心者にとっては非常に敷居が高いものとなっている。

我々は、誰もが簡単に３次元モデルを生成できるようにすることを目的として、手書きスケッチに基づくモデリング手法の開発を行っている [1]。本システム (通称 Teddy) では、ユーザは意図する３次元形状の輪郭を対話的に画面上に描いていくだけでモデルを生成することが可能であり、制御点を一つずつ移動したり、編集コマンドを組み合わせて使用する必要はない。図１に本システムを利用したモデリング例を示す。

スケッチに基づく３次元モデル生成システムとしては Brown 大学の SKETCH システム[4]が有名であり、それ以外にもいくつかの研究例が報告されている。これらのシステムが建物や機械部品といった人工的な物体のモデリングを対象としたものであるのに対し、我々のシステムは動物やぬいぐるみといった自由曲面によって構成された物体の生成を対象としている点を特徴とする。本システムによって作成された３次元モデルの例を図２に示す。

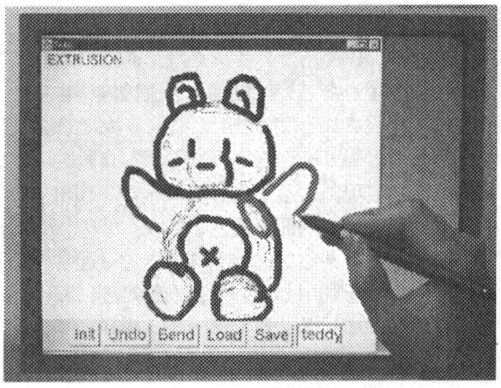


図 1. Teddy の使用例
対話的に輪郭を描いていくだけで３次元モデルを作成することができる。出来上がった３次元モデルは手書き風にレンダリングされる。

Teddy は SKETCH システムと同様に、簡単

"Teddy: A Sketch-based Modeling System", Takeo Igarashi (Univ. of Tokyo), Satoshi Matsuoka (Tokyo Inst. of Tech.), Hidehiko Tanaka (Univ. of Tokyo)

な概形を手軽に生成することを目的としており、正確な形状を時間をかけて生成する作業には適さない。このような性質を反映させ、ユーザの作業を円滑にするため、Teddy では手書き風レンダリング [2]を使用している。

本手法の応用例としては、通常の CG アニメーション用のモデラーとして利用の他、初心者のための娯楽用ソフトウェアとしての利用、正確なモデルを生成する前段階のプロトタイピングでの利用などが考えられる。また、医師が患者に疾患部位を説明したり、学校の先生が生体の構造を説明する場合など、電子黒板等を利用した環境でのコミュニケーション支援への利用なども考えられる。

本稿では、Teddy におけるモデリング操作について述べ、ついでアルゴリズムについて簡単に解説を加える。現在の実装では一般的なポリゴンメッシュによる表現を用いているが、内部表現としてはボクセルやメタボールといった表現を用いることも可能である。

Teddy は JAVA のアプレットとして WWW 上に公開されている。
www.mtl.t.u-tokyo.ac.jp/~takeo/teddy/teddy.htm

2. 関連研究

従来の 3 次元モデリングは、通常、ポリゴンを構成する頂点あるいは制御点を直接操作することによって実現される[2]。また、芯にあたる部分を指定して、その周りを包むような形状を自動的に生成する陰関数表現による手法も使われている[3]。

3 次元物体の内部表現としては、ボクセル[7]によるものなどがあるが、本システムでは通常のポリゴンメッシュを利用しており、内部処理においては[6]に紹介されているようなメッシュ操作アルゴリズムを利用している。

3. インタフェース

Teddy は基本的には液晶付きタブレットのようなペンベースの環境での利用を想定しているが、プログラム自体はマウスでも使用可能である。通常のモデリングソフトウェアがメニューやボタンを多用しているのに対して、Teddy でのモデリング操作はすべて手書きの自由曲線を画面上に描いていくことで進められていく。ペンのボタン(マウスの右ボタン)によってオブジェクトを回転することができ、この回転操作と描画操作によってモデリングが行われる。

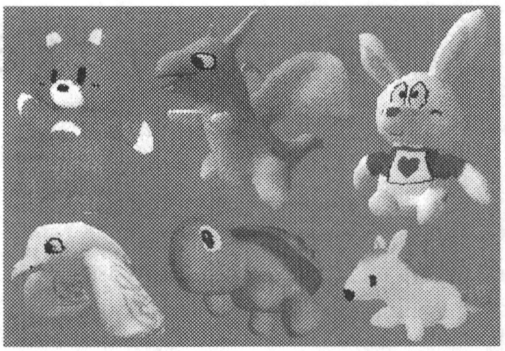


図 2. 作成された 3 次元モデルの例
このような形状を初心者でも数分で作成することが可能である。(テクスチャ作成および表示は市販ソフトウェアによる。)

4. モデリング操作の概要

図 3 にモデリング操作の概要を示す。まず、画面上に意図する 3 次元物体の輪郭を手書きの自由曲線によって描く(図 3a-b)。ペンが画面から離れると同時にシステムが対応する 3 次元形状を生成し、手書き風レンダリングによって画面に提示する(図 4c)。この時点で物体はすでに 3 次元形状として表現されているので、回転して視点を変えることができる(図 3d)。できあがった 3 次元物体に対しては手書きストロークによって様々な編集操作を加えることができる。物体上に線を描くことによって、表面への描画が行われる(図 3e-g)。ストロークがぐちゃぐちゃしていた場合には、消去ストロークと判断され、表面の線が消される(図 3h-j)。ストロークが閉じていた場合には、システムは自動的に「突起生成モード」に入る(図 3k-l)。この状態で視点を変更して(図 3m)、突起の輪郭に相当する線を描くと(図 3n)、突起が生成される(図 3o-p)。もし、ストロークが物体をまたがって引かれた場合には、物体がストロークによって切断される(図 3q-r)。切断操作はクリック操作によって完了することができる(図 3s)ほか、回転して輪郭線を引くことで切断面から突起を生成することもできる(図 3t-v)。オブジェクトの表面の突起やへこみを取り除き、滑らかにしたい場合には、突起生成モード中に消去ストロークを描く(図 3w-z')。

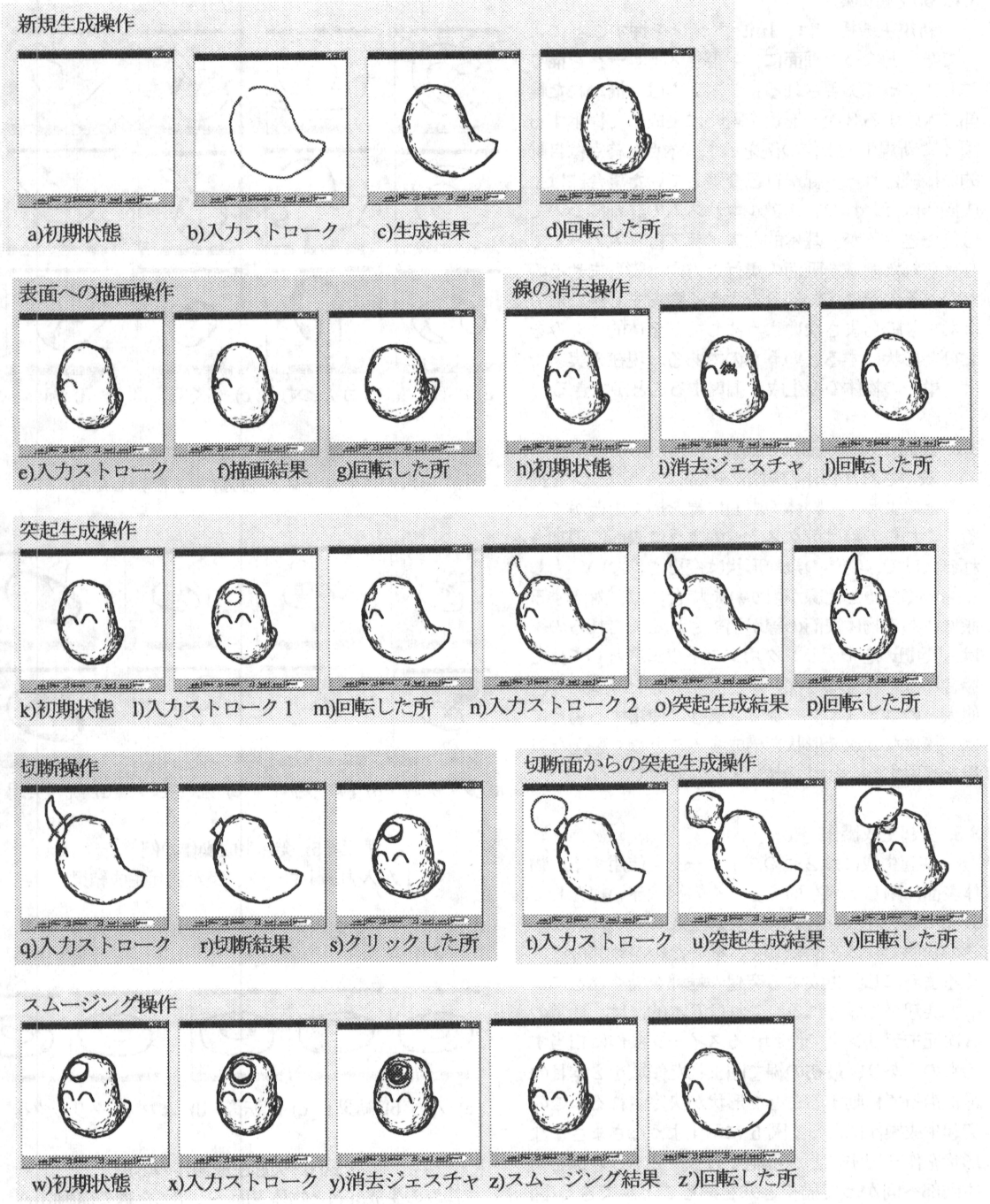


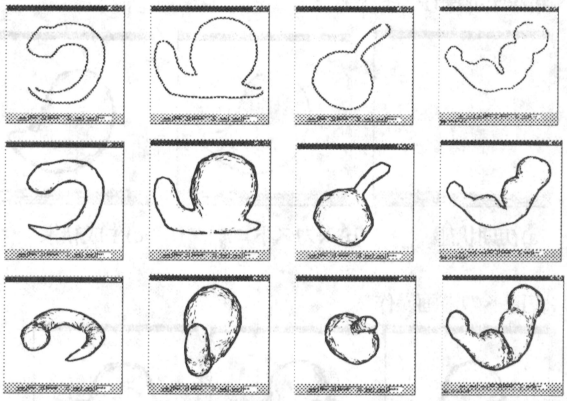
図 3: モデリング操作の概要

ストロークの描画とオブジェクトの回転の繰り返しによりモデリングが行われる。

e)。

3.1. 新規生成操作

新規生成操作は、Init ボタンを押すことによって空白となった画面に、手書きストロークを描くことによって実現される。システムは、描かれた輪郭に対応する３次元形状を瞬時に生成して提示する。図４に新規生成操作の例を示す。始点と終点は自動的に接続される。線が自己交差していた場合には生成操作は行われない。アルゴリズムの詳細については後で述べるが、基本的なアイデアは、入力として与えられた領域を画面の奥行き方向へ膨らませるのだが、その際に広い部分を大きく膨らまし狭い部分はあまり膨らまさないようにして、全体的に丸みをおびた形状にするというものである。現在の実装では、単一の物体のみ生成・編集することができる。

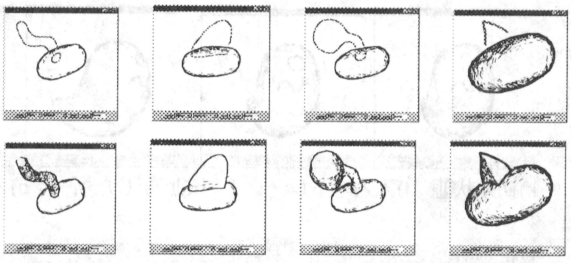


a) へび　b) かたつむり　c) さくらんぼ　d) 腕

図 4: 新規生成操作の例
上から入力ストローク、生成結果、回転した所

3.2. 表面への線の描画と消去

表示された物体の輪郭の内部にストロークを描くことによって、物体の表面に線を描くことができる。これらの線はテクスチャのように表面に追加されるだけで、物体の幾何形状は変更されない。もしぐちゃぐちゃした線が引かれた場合には、対応する領域にある物体表面の線が消去される。これらの線は、Teddy をモデリング用ソフトウェアとして見た場合には意味をなさないが、アイデアを練ったり、他人とのコミュニケーション支援に使用する場合には、複雑な３次元形状を構成することなく豊かな表現を実現することができ非常に効果的である。

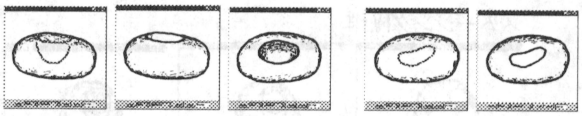


a) 長い突起　b) 薄い突起　c) 丸い突起　d) 鋭い突起

図 5: 突起生成操作の例
上が入力ストローク、下が突起生成結果

3.3. 突起生成操作

突起生成には２つのストロークを使用する。物体表面に閉じた線を描くと、その線は赤い線として表面に描かれ、システムは突起生成モードに入る。この状態で物体を回転して、描いた線が輪郭付近にくるようにし、生成する突起の輪郭を描くことによって突起が生成される。これは基本的には、通常の３次元モデリングで使われるスイープ操作に相当するものであり、最初の線で囲まれた領域を２本目の線に沿って移動することで形状が決定される。この突起生成操作によって図５に示すようなさまざまな形状を作り出すことが可能である。２本目の線を物体内部へ向かって描くと穴を掘ることができる(図 6a-c)。閉じた線を描いた後、突起を生成したくない場合には、一度クリックすると突起生成モードから抜けて、赤い線は通常の黒い線に変換される(図 6d-

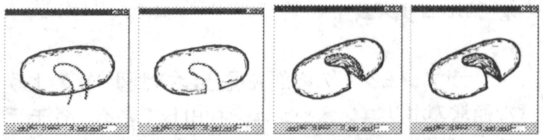


a) 入力 b) 結果t　c) 回転後　d) 入力 e) クリック後

図 6: 突起生成操作の応用
窪みの生成(a-c)と物体表面への閉じた線の描画(d-r)。

3.4. 切断操作

切断操作は物体をまたがるようなストロークを描くことによって実現される。その際、ストロークの進行方向向かって右側が残され、反対側は捨てられる。切断後は自動的に突起生成モードに入るので、一度クリックすることによって切断操作が完了する。この操作によって、物体を切り抜くといったことも可能である(図 7)。クリックするかわりに、２本目のストロークを描くことによって切断面上に突起を生成することもできる(図 8)。



a) 入力　b) 切断結果　c) 回転後　d) クリック後

図 7: 切断操作.

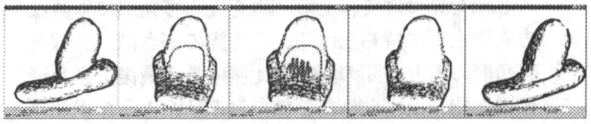
a) 入力　b) 切断結果　c) 回転後 d) 入力 e) 突起生成結果

図 8: 切断後の突起生成

3.5. スムージング操作

粘土で形を作るとき、余分な突起や窪みをなくして表面を滑らかにするような作業を行うことがある。これを行う操作がスムージングであり、突起生成モード中に、消去ストロークを描くことで実現される。この操作は物体表面の線を消す操作と異なり、実際に幾何形状を変更する。スムージングによって、余分な突起やへこみをなくしたり、突起生成時にできたエッジを滑らかにすることができる(図 9)。

a) 表面の凸凹の除去

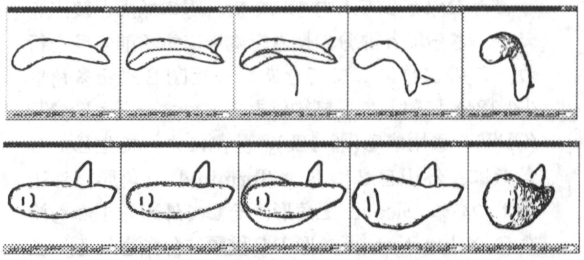


b) エッジを滑らかにする

図 9: スムージング操作

3.6. 変形操作

試験的な操作として、物体を大域的に変形する操作を実装している。まず、Bend ボタンを押して変形モードに入る。この状態で、まず変形の参照となるストロークを描く。ついで、その１本目のストロークがどのように変形されるかを示す、２本目のストロークを描く。すると、物体と１本目のストロークとの位置関係が、変形後の物体と１本目のストロークとの位置関係に対応するように変形が行われる。変形の影響は物体全体におよび、局所的な変形を行うことはできない。内部的にはポリゴンモデルを構成する各頂点の位置を画面に平行な方向にのみ移動している。変形操作によって、物体を曲げたり、伸ばしたり、膨らましたりすることができる(図 10)。



a) 操作前　b) 参照線　c) 対象線　d) 結果　e) 回転後

図 10: 変形操作の例
上が曲げの例で、下が膨らましの例

4. アルゴリズム

オブジェクトの内部表現としては図 11 のような通常のポリゴンメッシュを利用している。各モデリング操作によってメッシュが直接再構成され、CSG のような編集履歴による表現は用いていない。現在の実装では、球と同一のトポロジーを持つモデルしか扱うことができない。なお、以下に説明するアルゴリズムは計算速度向上のため、繰り返し計算を用いない直接的な計算方法に基づいている。将来的には、緩和計算等を用いたアルゴリズムを利用することによってより自然な形状を構成することも可能とえられる。本稿では、多くのモデリング操作のうち、もっとも特徴的な新規生成および突起生成についてのみ解説する。その他のアルゴリズムについては [1] を参照していただきたい。

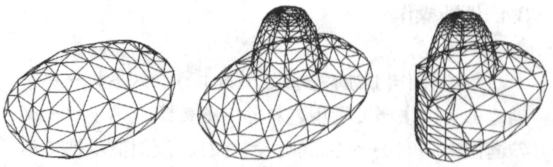


図 11: モデルの内部表現

4.1. 新規作成アルゴリズム

本アルゴリズムは、２次元の閉じた手書き曲線から３次元のポリゴンメッシュを生成する。大まかな流れとしては、まず、入力された２次元の多角形の芯線を見つけて、その芯線を元の多角形に垂直な方向に上下に持ち上げる。この際の高さは、芯線と多角形の周との間の距離に比例する。最後に多角形の周と持ち上げられた芯線を包み込むようなポリゴンメッシュを生成する。このような処理の結果、広い領域は大きく膨らみ、狭い領域はあまり膨らまず、全体として断面が楕円になるような３次元形状が作成される (図 12)。

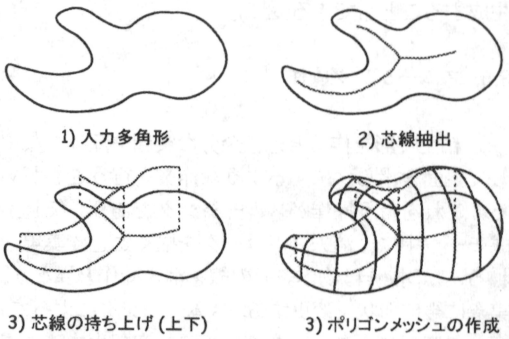


図 12: 新規生成アルゴリズムの概要

芯線は以下のようにして構成される [3]。まず、手書きの線として入力された折れ線の始点と終点を結び、さらに各線分の長さを均一にする前処理を行う。このようにしてできあがった閉２次元多角形 (図 13a) に対して、制約付きのドロネー三角形分割 (CDT)を適用する(図 13b)。分割によって生じた３角形は、外辺を２つもつ Terminal 三角形、外辺を１つもつ Sleeve 三角形、そして外辺を１つも持たない Junction 三角形の３種類に分類される。ここで内辺の中点を結んでいくことによって芯線の１種である Chordal Axis が生成される(図 13c)。しかしこのままでは膨らます操作に適さないので、芯線の枝刈りを行い、図 13 d,e を得る。

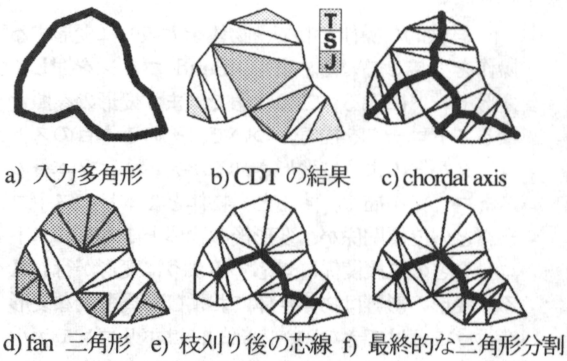
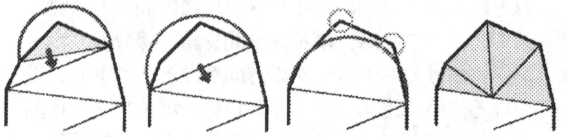


図 13: 芯線の構成

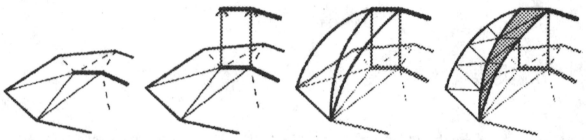
枝刈りは、各 Terminal 三角形から始めて内側へ進みながら、停止条件を満たす内辺が見つかるまで、内辺を順に取り除いてくことによって行われる(図 14)。停止条件とは、対象とする内辺と Terminal 三角形の間にある頂点のうち１つ以上が、その内辺を直径とする円の外にある状態をさす(図 14c)。すべての頂点が円内にあった場合には、その内辺は不必要と判断されて捨てられ、処理はより内側にある次の内辺へとすすむ(図 14a-b)。停止条件にあう内辺が見つかった場合には、その内辺の中点と各頂点を結ぶことによって Fan 三角形を生成して枝刈りを停止する(図 14d)。

以上の枝刈り処理によって図 14e のような三角形分割と芯線が得られる。最後に残った４辺形を三角形分割してから(図 13f)、芯線の持ち上げと面の生成を行う。その様子を図 15 にしめす。この時、持ち上げる高さは、芯線上の各点と、そこから外周にむかって伸びる内辺の長さの平均に比例する。芯線を持ち上げた後は、内辺が楕円の４分１をなすように持ち上げて、最後に三角形ポリゴンによる面張りを行う。



a) 内辺除去　b) 内辺除去　c) 枝刈り停止　d) fan 三角形

図 14: 枝刈りと fan 三角形の生成



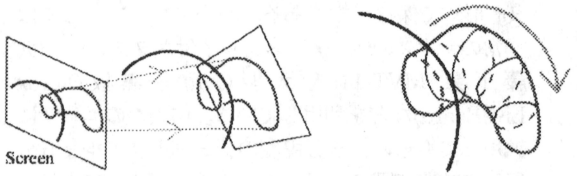
a)　処理前　b) 芯線持ち上げ　c) 内辺持ち上げ d) 面張り

図 15: 芯線持ち上げと面張り

4.2. 突起生成アルゴリズム

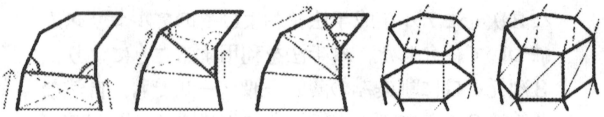
本アルゴリズムは、オブジェクト表面に描かれた閉じたストロークと、突起生成方向に描かれたストロークをもとに、突起に相当するポリゴンメッシュを生成する。概略としては、２本目のストロークを、オブジェクト表面に垂直な平面に投影して３次元形状としたあと、その投影された曲線にそってオブジェクト表面に描かれた閉多角形 m をスイープすることによってメッシュを生成する (図 16)。

投影面は以下のようにして決定される。まず、オブジェクト表面の閉多角形 m に垂直なベクトル v を求める[1]。さらに、このストロークの重心 g から視点方向に伸びるベクトル c を求める。投影面は、重心 g を通りベクトル v に平行な平面のうちで、もっともベクトル c に垂直に近い平面として得られる。



a) ストロークの投影　　b) スイープによるポリゴン生成

図 16: 突起生成アルゴリズムの概要



a) ポインタの移動　　b) メッシュ生成

図 17: オブジェクト表面の多角形のスイープ

[1] ベクトル v の計算方法は以下の通り。閉多角形 m 上の点を絶対座標系の XY 平面へ投影し、符号付きの面積を次の計算式によって計算する。

`Axy = 0.5*sum(i=0, i=n-1, x[i]*y[i+1]-x[i+1]*y[i])`

(ここで `x[n]=x[0]`). 同様にして Ayx と Azx を計算して得られるベクトル `v=(Ayz,Azx,Axy)` が求めるベクトル v である。

スイープ操作は、オブジェクト表面の閉多角形mを投影された2本目のストロークに沿って移動させていくことによって実現される。具体的には以下のようにして行われる。投影されたストロークの始点と終点から2つのポインタを順に内側へと移動させていく。この時、ポインタ間を結ぶ線分と、ポインタにおけるストロークの方向がなるべく垂直に近くなるように進めていく(図 17a)。ポインタの移動にあわせて、ポインタの対にあうように、多角形mと相似な多角形を生成していく。最後に隣り合う複製された多角形の対応する頂点を結び合わせることによってメッシュを生成する(図 17b)。この時、mの内部にあった、元のポリゴンは完全に取り除かれる。

5. 実装

プロトタイプシステムは Java™プログラム(JDK 1.1)として実装されている。Java3D™などは利用していない。コードサイズは約 13,000 行程度である。ポリゴンメッシュの生成は瞬時に実現される。モデルが複雑になってくると動作が遅くなるが、簡単なモデルを扱っている限り速度的なストレスは感じない。作成したモデルは Alias Wavefront の obj 形式で保存可能である。ハードウェアとしては通常のマウスの他、ディスプレイ付きタブレット(武藤工業 MVT-14) や大型の電子黒板(Xerox LiveBoard)などを利用している。何人かのユーザに試用してもらい、5分程度のチュートリアルと5分程度の練習で図2に示したようなモデルを自由に生成できるようになることを確認している。

6. おわりに

本稿では、手書きストロークを利用した対話的な3次元モデル生成手法、およびそのアルゴリズムについて紹介した。本手法を利用することにより、3次元CGに馴染みの薄い一般ユーザでも、自然な曲線的3次元モデルを短時間に作成することが可能となる。

今後の発展としては、アルゴリズムの改良による形状表現能力の向上、テクスチャ編集などの機能拡張、およびアニメーションへの対応などがあげられる。アルゴリズムに関しては、特に突起生成アルゴリズムの拡張が必要であり、メタボールのような陰関数の考え方を利用した方法を検討している。機能拡張に関しては、テクスチャ編集の他にコピーなどの編集機能の付加、他のモデリングソフトウェアとの連携の強化などが挙げられる。自動的に左右対称な形状を生成する機能なども有用と考えられる。アニメーションについては、突起生成操作時に自動的に階層構造を付与することにより、胴体に付け加えた頭や腕を回転することができる拡張版が実装されており、CMUで開発中の3次元アニメーションシステム Alice [5] とともに広く配布されている。

謝辞

本研究は、文部省科学研究費補助金(課題番号10-04686)による研究成果の一部である。

参考文献

1. Igarashi,T., Satoshi,M., Hidehiko,T., "Teddy: A Sketching Interface for 3D Freeform Design", *in Proc. of SIGGRAPH 99*, 409-416, 1999.
2. MacCracken, R. and Joy, K.I. Free-form deformations with lattices of arbitrary topology. *in Proc. of SIGGRAPH*, pp. 181-188, 1996.
3. Markosian, L., Cohen, J.M., Crulli T. and Hughes, J.F., Skin: A Constructive Approach to Modeling Free-form Shapes. *in Proc. of SIGGRAPH 99*, 1999.
4. Markosian, L., Kowalski, M. A., Trychin, S.J., Bourdev, L., Goldstein, D., Hughes, J.F., "Real-Time Nonphotorealistic Rendering", *in Proc. of SIGGRAPH 97*, pp. 415-420, 1997
5. Parasad, L., "Morphological Analysis of Shapes", CNLS newsletter, No.139, July 1997.
6. Taubin, G. A signal processing approach to fair surface design. *in Proc. of SIGGRAPH 95*, pp. 351-358, 1995.
7. Wang, S.W. and Kaufman, A.E., Volume sculpting. *1995 Symposium on Interactive 3D Graphics*, pages 109-116, 1995.
8. Zeleznik,R.C., Herndon,K.P., Hughes,J.F., "SKETCH: An Interface for Sketching 3D Scenes", *in Proc. of SIGGRAPH 96*, pp. 163-170, 1996.
9. Alice, Carnegie Mellon University, 1999, http://www.alice.org