2006年度　卒　業　論　文

簡易な形状の俯瞰と形状変形を目的とした3Dモデリング手法とインタフェースの提案

指導教員：渡辺　大地講師

メディア学部 ゲームサイエンスプロジェクト
学籍番号　M0103190
斉藤　亜紀子

2006 年度　　卒　業　論　文　概　要

論文題目

簡易な形状の俯瞰と形状変形を目的とした 3D モデリング手法とインタフェースの提案

メディア学部 **学籍番号：M0103190**	**氏名**	斉藤　亜紀子	**指導教員**	渡辺　大地講師

キーワード	3DCG、モデリング、インタフェース、 ジェスチャー、多面図

3DCG の分野は近年急激な発展を遂げている。多くの分野で様々な技術が用いられており、今後も更なる発展が期待される分野である。それに伴い 3DCG のモデリングを行うアプリケーションも数多く提供されているが、そのほとんどは複雑なコマンドの理解や技術の習得が必要であり、厳密なモデリングが可能であるがモデラーの操作に慣れていない人や単純なモデリングを行う場合には適さない。それらのケースで使用するモデラーはモデルの把握が容易なインタフェースとその上で手軽に行える操作形態が必要である。

手書きの入力を基にし、3D モデルへの変換を行うモデリング手法はいくつか存在している。代表的なものはマウスの特定のジェスチャーをあらかじめ 3 次元形状と対応付けておき、ユーザのストロークから 3D モデルを構築する「SKETCH」や、ユーザの二次元のストロークから形状の大きさを算出して 3D モデルを生成する「Teddy」などがある。しかし、これらは操作は容易なものでありそのシステムが予測出来る形状の範囲内であれば簡単なモデリングが可能であるが、生成出来るモデルが限られていたり、モデルの配置、俯瞰が少々困難であるという問題点などがあげられる。

本研究ではモデリング手法としてポインティングデバイスを用いた手書きの入力、ジェスチャーベースのモデリングと、一画面でモデルの全ての面を俯瞰するキューブ型の多面図、それを実現したツールを提案する。

目 次

第 1 章　序論 **1**

第 2 章　3DModelingTool”Dice” **5**
2.1 概要 5
2.2 モデルの生成 6
2.2.1 描画に用いるストローク配列の定義 6
2.2.2 形状モデルの新規生成 6
2.2.3 投影モデルの生成 8
2.3 モデルの編集 9

第 3 章　評価 **10**
3.1 基本操作 10
3.2 モデリング事例 15
3.3 ユーザからの評価 18

第 4 章　問題点と展望 **19**
4.1 問題点 19
4.2 展望 20

第 5 章　結論 **21**

謝辞 **22**

参考文献 **23**

図 目 次

3.1 ユーザの描画ストローク 11
3.2 形状モデルと投影モデルの生成 11
3.3 投影モデルの頂点のピック 12
3.4 投影モデルの頂点の移動 12
3.5 モデルを上方から見た図 12
3.6 モデルを奥側から見た図 12
3.7 投影モデル表示のスイッチが ON 13
3.8 投影モデル表示のスイッチが OFF 13
3.9 モデルの回転・平行移動 13
3.10 別角度から見た図 13
3.11 Dice に備えた 12 色 14
3.12 別角度からの参照 14
3.13 キノコの傘の部分 15
3.14 キノコの配置 15
3.15 1UP キノコできました 16
3.16 円柱と円錐で作成した鉛筆 17
3.17 鉛筆を上から見た図 17
3.18 直方体で作成した家や庭 17
3.19 別角度から見た図 17

第 1 章

序論

近年、急激に開発が進んでいるのが 3 次元においてのコンピュータグラフィクスである。多くの分野で様々な技術が用いられており、今後も更なる発展が期待される分野である。それに伴い 3 次元コンピュータグラフィクス用のモデリングツール（以下 3D モデラーと呼ぶ）も数多く提供されている。それらは個人向けに無償で提供されているものから、高度な技術を用い極めて精度の高いモデリングを行う企業や開発者に向けたものまで様々である。

一般的な 3D モデラーは X,Y,Z の 3 軸が予め表示されたウィンドウに対し、もともとサポートされているプリミティブな図形をユーザ側で選択し、それを僅かずつ編集してゆくことでモデルを形成する形態が多い。また 3D モデラー内においてのモデルの形状や座標の把握には、1 つから 4 つのウィンドウを表示し、各画面で視点を回転させることによって行う。複数の画面の場合にはウィンドウを分割し、例えば正面、後方、上からなどモデルを多方向からそれぞれ独立した別の画面で見ることになる。

しかし多くの 3D モデラーは煩雑なコマンドからなるインタフェースを備えており、ある程度の意図的、意匠的なモデリングを行うにはそれらのコマンドの理解や技術の習得が必要不可欠となる。また直感的なモデリングが行い難い要因として、3 次元空間においてのモデルの把握が容易でないことが挙げられる。一画面でモデルを見る場合でも多画面でモデルを見る場合でも、モデルの全体図の把握の困難

さが、意図しない編集結果を生み出すことが多くある。3 次元でのモデリングに不慣れなユーザはうまくモデルと各座標の位置関係を捉えることが出来ないからである。初心者に向けたモデラーにはモデルの現状が把握しやすいインタフェースと、その上で手軽に行える操作形態が必要である。

直感的モデリングを行う、という目的で開発された 3D モデラーはいくつか存在し、その中でマウスの動きから 3 次元形状を構築してゆくジェスチャーの手法に V.Branco[1] の研究や Robert C.Zeleznic の “SKETCH” [2] がある。これは特定のマウスの動きと生成する 3D モデルを対応付けておき、それに添ったユーザの 2 次元の手書きのストロークから 3 次元形状を構築するもので、モデルの移動や拡大縮小、伸縮、コピーなどがある程度自在に、かつ迅速に行うことが可能である。プリミティブ形状も多数備えており、入力のほとんどがマウスによるジェスチャーなので単純なインタフェースで多くの形状を生成することが出来る。生成した各モデルの配置は画面内の既存のモデルとの位置関係が基になり、またモデルに影を付け足すことでそのモデルは「浮いた」状態になるなど、それらの情報を駆使することで違和感のない配置が可能である。しかしこれらは先に述べたように予め備えられている形状を配置してゆくに過ぎず、例えば都市や部屋のデザインなど、ある程度の決まった図形を空間に配置していく操作においては非常に優れたシステムであるが、ユーザの入力を直接反映するものではない。ストロークに対応付けられた 3D モデルのみの生成で、自由曲面を持つ形状に関しては入力が難しいという問題点がある。

手書きによるスケッチを基に三次元形状を生成する研究は様々なもの [3]〜[10] がある。これらは用いている手法はそれぞれ異なるが、予めユーザが行っておいたスケッチ図を読み込み 3D として生成するものである。ユーザが描いた二次元でのデッサン図から 3D モデルを構築するが、リアルタイムにモデリングを行うものではなく、デッサンの狂いなどから発生する形状の歪みや、汎用的な形状の生成には対応していない。

また、同じくストロークの入力から 3 次元形状を表現する手法はいくつか存在

するが [11][12][13][14]、その中に五十嵐健夫の “Teddy” [15] がある。ユーザは手書きで閉じた曲線をウィンドウに直接描画し、システムがそのストロークからユーザが意図する形状を予測し、3D モデルとして生成する。この場合奥行きは入力した曲線から自動的に算出し、適用する。ユーザは絵を描く感覚で簡単に自由曲面からなる 3 次元のモデルを作成することが可能である。インタフェースも数個のコマンドを GUI 上に備えているだけであり、初心者に扱いやすく直感的な操作が可能である。このシステムを基にプレイステーション 2 用のゲームソフト「ラクガキ王国」[16] や、e-frontier 社の 3D モデラー「マジカルスケッチ」[17] が開発され、いずれもユーザから高い評価を受けている。しかし Teddy においてもそのシステム自体が最初から意図している形状、たとえばぬいぐるみの様な自由曲面を持つ簡単な形状を作成する点においては単純な操作でユーザの意図を正確に反映することが出来るが、Teddy が意図していない形状、たとえば立方体や多角柱などの角ばった形状や鉛筆の芯の様な尖った形状を生成するのは実質的には不可能である。これは Teddy がもともとそのような形状を作成することを念頭に置いていないからである。また Teddy では生成するひとつひとつのモデルが独立しているわけではなく、大元のモデルに付け足していく形でモデルを形成してゆくが、モデルの配置はシステムが適切な位置を自動的に決定するため、ユーザの意図通りにならないケースがしばしば起こる。そして編集中は基本的にユーザが平面でモデルを捉えることを前提としているため、モデル同士の位置を 3 次元的な視点で確認することは難しい。

本研究では、上記の問題点を踏まえ、簡単な操作でモデルを生成出来る技法と、モデルを容易に俯瞰することが可能であり、操作に特別な知識の必要のないインタフェースの両方を備えるシステムを実現した。ユーザがそのシステムを扱う際、モデルに対して行うオペレーションを出来る限り簡易なものに留め、モデルの位置、角度、大きさなどの要素をユーザが容易く知覚出来るような手法を提案する。

具体的には、ユーザはモデリングを行う際、まず手書きでプリミティブな形状を生成する。本システムでは、形状を画面内にあらかじめ用意されたキューブ状

（正六面体）の内部に生成し、形状の右側の稜線、左側の稜線、上側の稜線、奥側の稜線がそれぞれ隣接しているキューブの平面に投影する。それぞれの面に投影した稜線を個別にユーザが編集することで形状が変形する。これによりユーザは常に作成する形状のそれぞれの面を認識しながらモデリングを行うことが出来る。またこの方法であればユーザが頂点のひとつひとつを直接編集するため、曲面を自由に変形することが可能となる。

本論文の構成は以下の通りである。2 章では研究の成果である 3D モデリングシステムの概要、機能の詳細や実装について述べる。3 章で評価を行い、4 章で問題点と展望を述べる。5 章で本研究の結論を示す。

第 2 章

3DModelingTool”Dice”

本章ではツールの概要、理論と実装を述べる。

なお、以降ではこのツールを”Dice”と呼ぶ。この名称は以降に記すツール独自のインタフェースに由来するものである。

2.1 概要

Dice においてはモデリングを基本形状の生成とその変形操作の二段階で行っていく。基本形状の生成は、ポインティングデバイスのストロークによる軌跡から三次元形状を生成する方法を取る。ストロークから生成するモデルはもともと Dice がサポートしているプリミティブなものであり、ユーザはそれらのモデルを徐々に変形させていくことで意図に沿った形状を構築する。用意している形状は、球、円柱、円錐、直方体である。生成された形状を以降は形状モデルと記す。その大きさや生成位置の決定方法は次節以降で述べる。

形状の変形操作は、形状に対して直接位相要素をユーザが編集するのではない。生成の際同時にシステムは予めインタフェースとして生成しておいた、モデルを囲むキューブにそのモデルの全ての頂点と稜線を上部、右方、左方、奥のそれぞれの面にワイヤーフレームで投影する。それらの投影は形状モデルと対応付けられており、ユーザはその投影を編集することが可能である。それにより、頂点や

稜線がそれぞれに対応する形状を部分的に編集する。生成された投影を以降は投影モデルと記す。

このキューブは多面図の役割も果たす。モデルの生成や編集はこの内部で行うことを前提とするが、これによりユーザは形状モデルの位置、角度、モデル同士の配置の把握が容易に行え、形状モデルの視覚的な情報を一度に俯瞰することが可能となる。ユーザはモデルのそれぞれの面を参照し、必要に応じてウィンドウからの視点と形状モデルをそれぞれ回転させながら投影モデルを編集し、形状を構築する。投影モデルの操作に関する詳細は次節以降で述べる。

2.2 モデルの生成

2.2.1 描画に用いるストローク配列の定義

ユーザはストロークを画面に引くことで形状モデルを生成する。ストロークの軌跡は視線ベクトルに垂直で原点を通る面を投影面とし、一定の距離ごとに線分に頂点を生成している。このストロークによって生成する頂点の位置ベクトルの配列を $\{\mathbf{M}_0, \mathbf{M}_1, ..., \mathbf{M}_n\}$ と定義し、うち任意要素を $\mathbf{M}_i$ と記す。

円錐、円柱、直方体を生成するとき、入力としてL字を描くストロークを必要とする。その際、縦方向のストロークの長さが形状モデルの高さ、横方向のストロークの長さが底面もしくは幅の長さになるが、このときのL字の角部分の頂点の位置ベクトルを $\mathbf{M}_k$ とする。なお、$\mathbf{M}_k$ は $\mathbf{M}_0$ から $\mathbf{M}_n$ までの頂点でそれぞれ前後の線分の方向ベクトルの内積を比較し、最も値が大きかったものとした。

このストローク配列を用いた形状モデル生成の詳細は次節で述べる。

2.2.2 形状モデルの新規生成

ユーザがモデルを作成する際の手順とその理論を記す。

まずユーザは一番初めにプリミティブなモデルを生成する。ユーザはDiceに備えられている基本形状生成用のチェックボックスに自分が作成する形状にチェック

を入れてから手書きのストロークで形状の半径、高さ、幅を指定する。

以下に生成出来る基本形状の一覧と形状ごとのストロークの方法、形状ごとに用いる値の算出方法、二次元平面状においてのストロークの軌跡を基に形状を3次元空間内に生成するための座標値を記す。用いる理論は球、円柱・円錐、直方体でそれぞれ異なるため別に表記した。

球の生成時、画面に1ストロークで円を描きその半径を形状モデルの半径に適用する。以下は半径 R を求める式 (2.1) と生成した球の中心の座標値 $\mathbf{P}$ を求める式 (2.2) である。

ストロークの始点のベクトルは $\mathbf{M}_0$、終点のベクトルは $\mathbf{M}_n$ と記す。

$$R = \frac{|\mathbf{M}_{\frac{n}{2}} - \mathbf{M}_0|}{2} \tag{2.1}$$

$$\mathbf{P} = \frac{\mathbf{M}_0 + \mathbf{M}_{\frac{n}{2}}}{2} \tag{2.2}$$

円柱・円錐の生成時、画面に1ストロークでL字を描きその縦線の長さを高さ、横線を長さを底面の半径に適用する。以下は底面半径 R を求める式 (2.3)、高さ H を求める式 (2.4)、形状の中心の座標値 $\mathbf{P}$ を求める式 (2.5) である。なお、円柱、円錐の形状の中心は底面の中心とする。

$$R = \frac{|\mathbf{M}_n - \mathbf{M}_k|}{2} \tag{2.3}$$

$$H = |\mathbf{M}_k - \mathbf{M}_0| \tag{2.4}$$

$$\mathbf{P} = \frac{\mathbf{M}_k + \mathbf{M}_n}{2} \tag{2.5}$$

直方体の生成時、画面に1ストロークでL字を描きその横線を長さを幅に、縦線の長さを高さに適用する。以下は幅Wを求める式(2.6)と高さHを求める式(2.7)と形状の中心の座標値Pを求める式(2.8)である。なお、直方体の形状の中心は重心とする。

$$W = |\mathbf{M}_n - \mathbf{M}_k| \tag{2.6}$$

$$H = |\mathbf{M}_k - \mathbf{M}_0| \tag{2.7}$$

$$\mathbf{P} = \frac{\mathbf{M}_0 + \mathbf{M}_n}{2} \tag{2.8}$$

2.2.3 投影モデルの生成

生成した形状モデルの頂点と稜線はウィンドウ内のキューブの上側、右側、左側、奥側の面に投影する。この投影モデルの生成は単純に形状モデルの頂点のデータと稜線のデータを設定したキューブの面に射影してゆく。4つの投影面において、それぞれの式は上側が$y = 50$、右側が$x = 50$、左側が$x = -50$、奥側が$z = -50$となる。

一般的には3次元形状のデータとモデルの姿勢、位置ベクトルは別に管理する。それを反映する変換行列によって演算した結果を表示する。ここではその変換行列を$\mathbf{M}$とおく。また、求める投影モデルの頂点の位置ベクトルを$\mathbf{P}$、元形状データ中の頂点の位置ベクトルを$\mathbf{B}$とする。このときその頂点が実際に表示されている位置は$\mathbf{MB}$となる。従ってキューブに投影するための正射影変換をHとすると、投影は以下の式(2.9)で求めることができる。

$$\mathbf{P} = H\,(\mathbf{MB}) \tag{2.9}$$

モデル行列 $\mathbf{M}$ は実際には平行移動と回転移動の合成変換であり [19]、これにより位置を決定してゆく。モデルの姿勢を $\mathbf{R}$、平行移動を $\mathbf{T}$ とするとき $\mathbf{M}$ は次の式 (2.10) で表す。

$$\mathbf{M} = \mathbf{TR} \tag{2.10}$$

$\mathbf{T}$ や $\mathbf{R}$ の値が変われば $\mathbf{M}$ の値も変わる。再び式 (2.9) を適用することで投影モデルの再構築を行う。

2.3 モデルの編集

形状モデルに対しての変形操作は、作成した形状モデルに対して直接位相要素を編集するのではなく、投影モデルに対して行う。それぞれの投影モデルは形状モデルに対応しており、投影モデルを編集することで形状モデルもそれに追随する。また形状モデルの変形は他の投影モデルにも反映する。形状モデルの対象となる頂点に関し、形状モデル上での頂点の位置ベクトルを $\mathbf{B}$、投影モデル上での頂点の位置ベクトルを $\mathbf{P}$、変形後の形状モデルの頂点の位置ベクトルを $\mathbf{B}'$、変形後の投影モデルの頂点の位置ベクトルを $\mathbf{P}'$ とする。

モデル行列 $\mathbf{M}$ はユニタリー行列なので逆行列が必ず存在する [20]。一方で、正射影変換 H は行列で表記が可能だがその場合には逆行列が存在しない。従って行列演算とは別の手段で逆変換を定義する必要がある。具体的な方法として、投影面上での平行移動量を表示モデル上の移動量としてこれを実現した。これにより、以下の式で変換が可能となる。

$$\mathbf{B}' = \mathbf{M}^{-1} \cdot H^{-1}(\mathbf{P}') \tag{2.11}$$

以上により投影モデルから形状モデルにデータを反映する。

第 3 章

評価

本研究ではモデリングツール「Dice」を 3DCG ツールキットである FKSystem[18] を用いて実装し、実際にモデリングと検証を行った。本章では Dice の機能を解説しながら検証の結果を述べる。

3.1 基本操作

ユーザは生成したい形状のチェックボックスにチェックを入れてからストロークを行う。予め備えている基本形状は球 (Sphere)、円柱 (Prism)、円錐 (Cone)、直方体 (Block) である。

図 3.1 は Sphere にチェックを入れストロークをしているところである。

また形状モデルの生成と同時にキューブに投影モデルが現れる。図 3.2 は形状モデルを生成し、それに伴ってワイヤーフレーム状の投影モデルが生成されたところである。

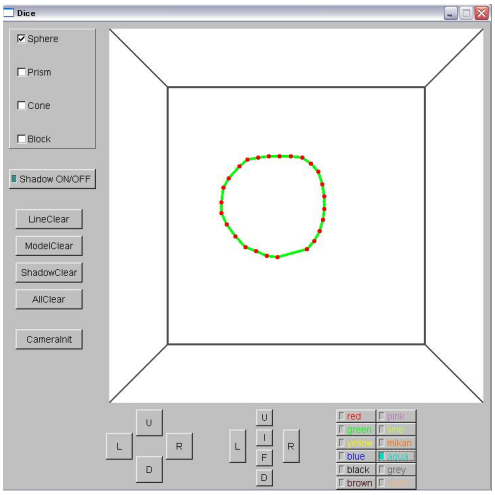

図 3.1: ユーザの描画ストローク

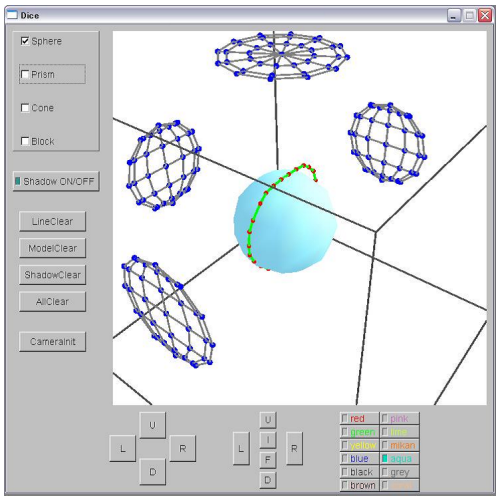

図 3.2: 形状モデルと投影モデルの生成

投影モデルは形状モデルに対応しており、各投影モデルの位相要素、Dice では頂点、稜線であるが、それらを編集することによって対応付けられている形状モデルもそれに追随して変化する。ユーザが直接的に動かすことが出来るのは投影モデルの頂点である。

Dice では投影モデルの頂点を移動するキーに右ドラッグを割り当てている。投影モデルにマウスポインタを近づけるとシステムが自動的に近傍の頂点をピックする。ピックした頂点は色が赤く変わる。ユーザは頂点を選択し、ドラッグすることで頂点とそれに付随する稜線が各面に沿って移動する。図 3.3 は頂点をピックしたところで、図 3.4 が実際に頂点を移動しているところである。投影モデルに沿って形状が変形するのが確認できる。図 3.5 と図 3.6 は図 3.4 をそれぞれ別の角度から見たものである。

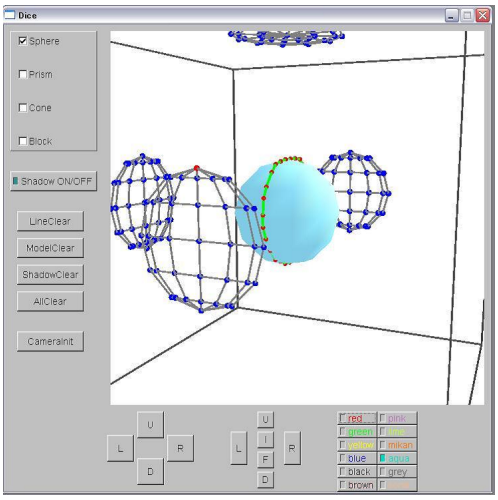

図 3.3: 投影モデルの頂点のピック

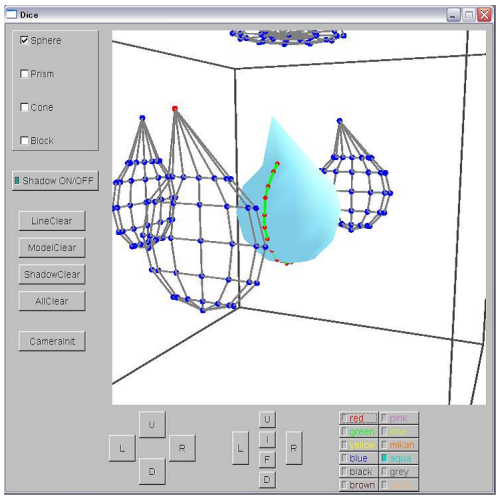

図 3.4: 投影モデルの頂点の移動

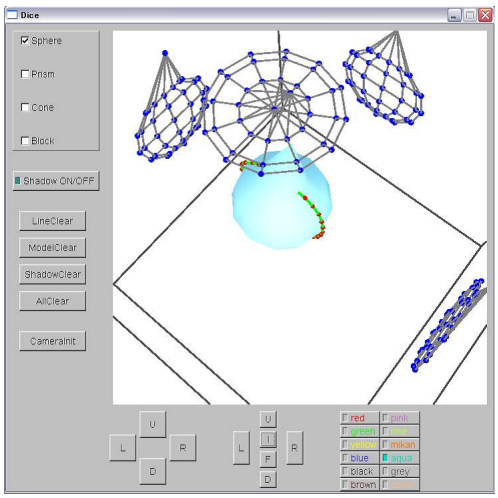

図 3.5: モデルを上方から見た図

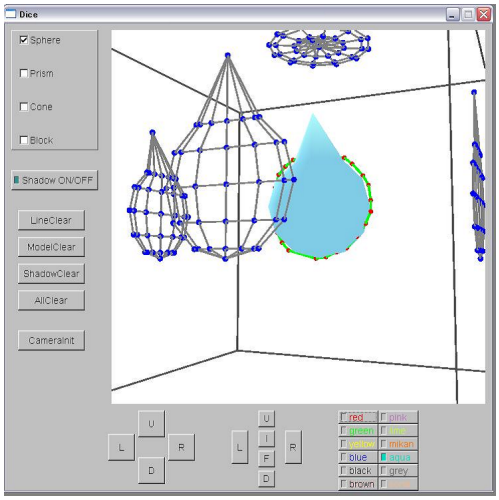

図 3.6: モデルを奥側から見た図

ユーザは「Shadow ON/OFF」スイッチで投影モデルを必要に応じて可視か不可視に切り替えることが出来る。また「LineClear」ボタンで描画ストロークを、「ModelClear」ボタンで最新の形状モデルと投影モデルを、「ShadowClear」ボタンで全ての投影モデルを、「AllClear」ボタンで画面内にある全ての形状モデルと投影モデルを消去する。キーボード十字キーでカメラの視点を操作し、「CameraInit」

ボタンはカメラの視点を初期値に戻す。図 3.7 は投影モデルが可視の状態、図 3.8 は投影モデルが不可視の状態である。

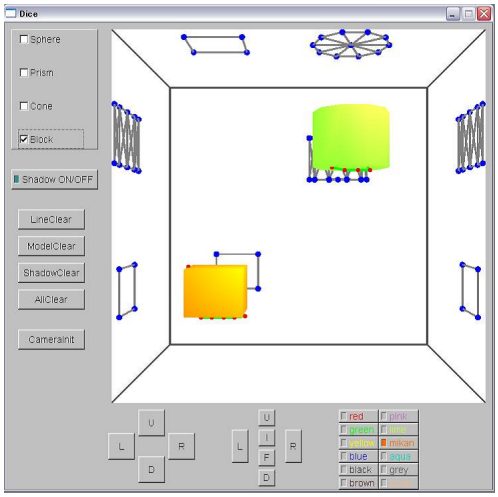

図 3.7: 投影モデル表示のスイッチが ON

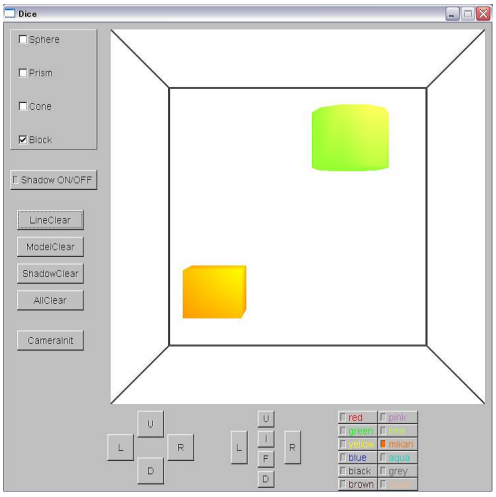

図 3.8: 投影モデル表示のスイッチが OFF

ユーザは画面下部の 4 つボタンで形状モデルの回転、6 つボタンで X,Y,Z 方向への平行移動を行う。投影モデルはそれに追随して変形する。図 3.9 は図 3.7 の直方体モデルを回転・平行移動したものである。またそれを別の角度から見たのが図 3.10 である。投影モデルが形状モデルに追随しているのが確認できる。

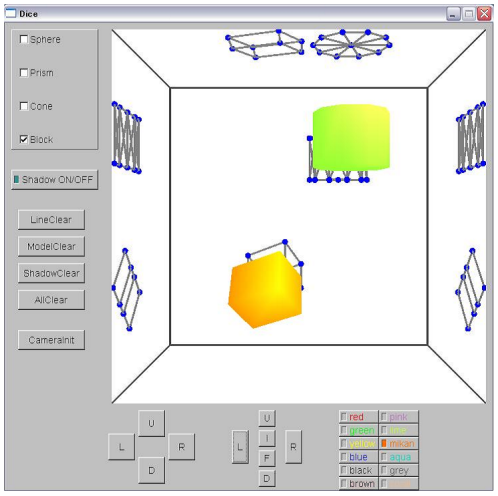

図 3.9: モデルの回転・平行移動

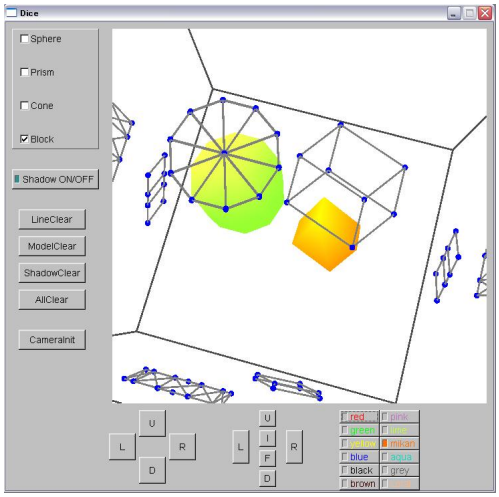

図 3.10: 別角度から見た図

形状モデルの色は生成時に右下部にあるカラーパレットが選択している色で決定する。現在 Dice がサポートしているのは、赤、緑、黄、青、黒、茶、ピンク、ライム、オレンジ、アクア、灰、コーラルの 12 色である。図 3.11 は実際に 12 色全てを使用したもので、図 3.12 はそれにカメラの視点を動かし投影モデルを消去したものである。

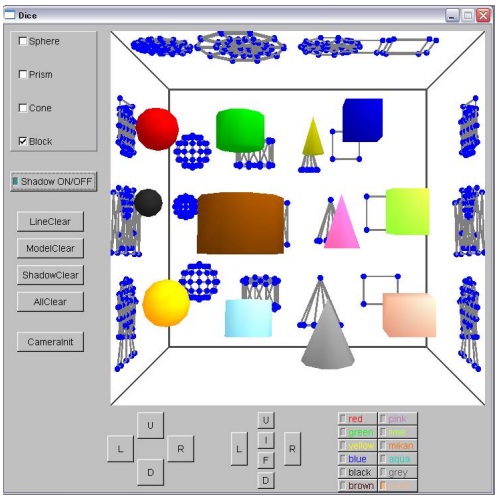

図 3.11: Dice に備えた 12 色

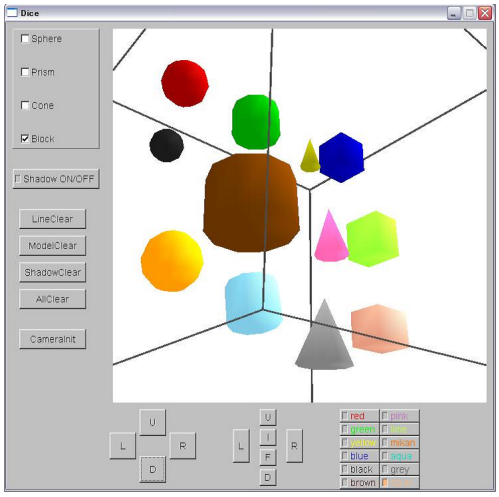

図 3.12: 別角度からの参照

3.2　モデリング事例

モデリングの事例として基本形状からの変形の例をいくつか示す。

図 3.13 のキノコはまず球を生成してから下側の頂点を移動することで傘の部分を作っている。図 3.14 で投影モデルを参照しながらヘタと傘の位置合わせを行う。図 3.15 が完成図である。なお、このキノコは 4 つの形状モデルから成るが、全て球の基本図形のみを用いて行った。

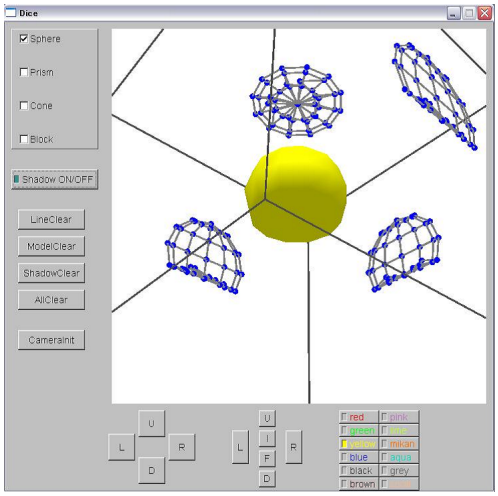

図 3.13: キノコの傘の部分

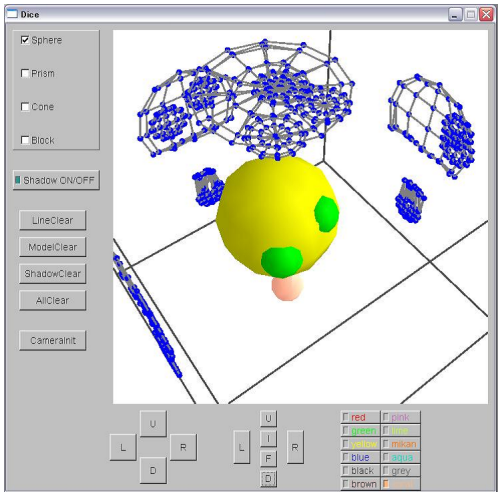

図 3.14: キノコの配置

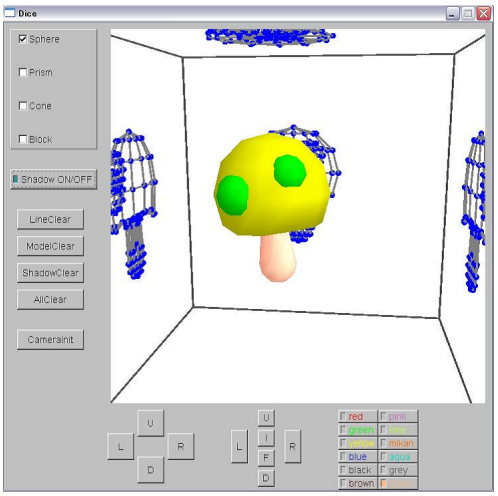

図 3.15: 1UP キノコできました

図 3.16 は円柱・円錐を編集して作成した鉛筆、図 3.17 はそれを上から見たものである。

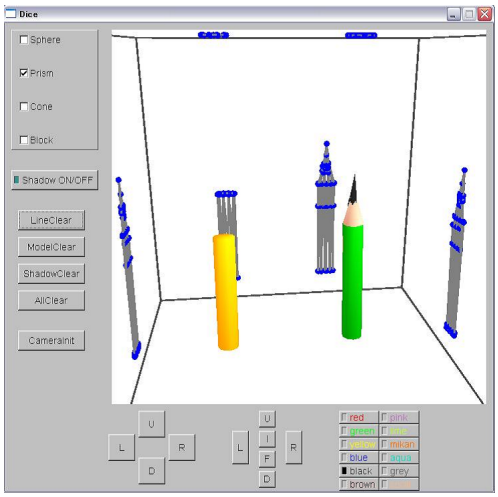

図 3.16: 円柱と円錐で作成した鉛筆

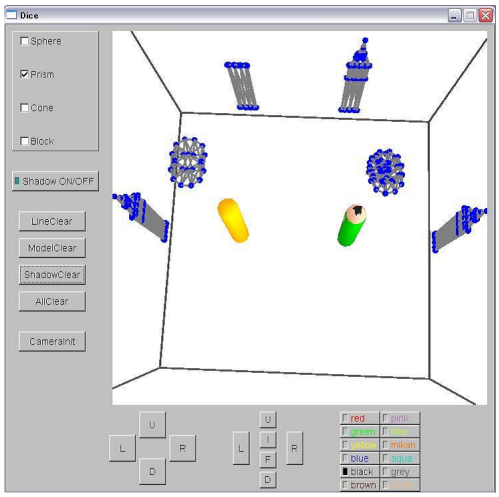

図 3.17: 鉛筆を上から見た図

図 3.18 は直方体を編集して作成した家や円錐から形成した木、図 3.19 はそれを別の角度から見たものである。

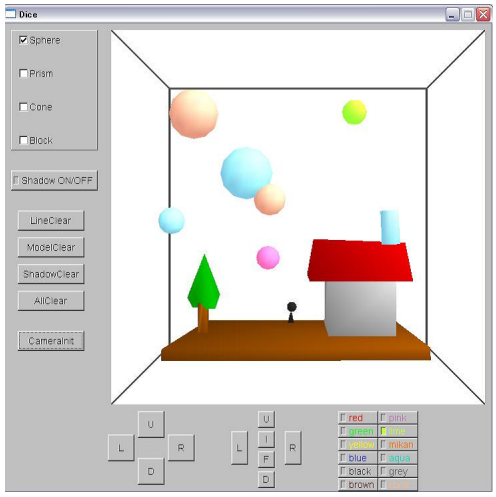

図 3.18: 直方体で作成した家や庭

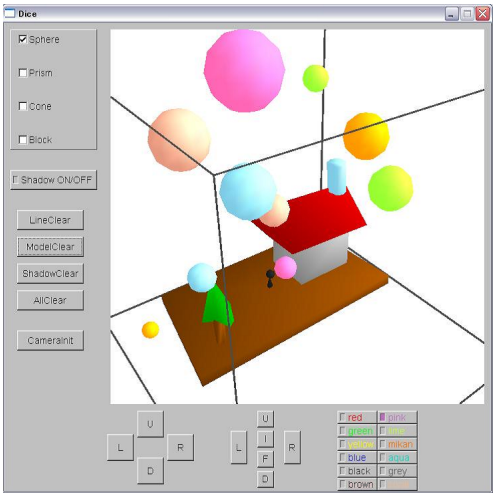

図 3.19: 別角度から見た図

いずれも投影モデルを効果的に利用することが出来た。

3.3 ユーザからの評価

実際に Dice を使用したユーザから以下の様な評価を得た。

GUI でモデルの移動・回転を行えるのが良い、投影モデルの存在により従来のものよりもモデルの配置がしやすい、もっとジェスチャーで様々なことが出来た方が良い、などである。ツールの完成度に起因する要望、例えば「Undo 機能がない」「モデルの保存機能がない」などの意見も少なからずあったが、投影モデルを使用したインタフェースに関しては肯定的な評価を得ることが出来た。

第 4 章

問題点と展望

4.1 問題点

本研究では実際にツールを実装し、モデルの投影操作という手法を用いることで簡易な形状のモデリングにおいては従来のモデリングツールと比べモデル全体の把握や配置、形状の変形を容易く行うことが出来たが、いくつかの問題点がある。

Dice では基本形状の生成をユーザのストロークから行い、その後投影モデルを基に変形していくが、そのためトーラスなど穴の空いたモデルを表現できない。また一度に生成する形状が多くなるほど投影モデルもそれに伴い増加するので、多量のモデルを生成し、どのモデルもテンポラリーに編集してゆく場合などにはあまり適さない。基本形状の分割数が小さく、投影する頂点や稜線が少ないので微細なモデルを生成する場合にも適さない。

投影を生成する平面をキューブという形で表現したため、キューブ内の範囲でしかモデリングが行えない。

投影を編集する際、頂点のみをピックし稜線はそれに追随しているが、追随する稜線が滑らかな弧を描いたり、また稜線のみの編集は行うことが出来ない。

4.2 展望

本研究とそれに伴い実装したモデリングルール Dice では、ユーザのストローク入力からによる形状の生成と投影の編集による簡易なモデリングを実現したが、ツールそのものの完成度が高くなく、やや操作性に問題があるものとなった。

色の選択の自由度を高くする、新規生成のための基本形状を増やす、形状モデルごとの投影の色を変え投影モデルの区別をし易くする、一般的なツールでは当たり前に備えられている「形状のコピー、ペースト」「一つ前の操作をやり直す」「形状モデルに対しての個別の操作」「頂点・稜線の複数選択」など本ツール自体、改善の余地は随所に存在している。これら等の様々な機能を付加することで、Dice はより使いやすく、本研究の目的とより合致するものになるだろう。

また、一般の 3D モデリングは通常リアルタイムで行うものなので、Dice においてもシステム内の処理速度を重視し、形状データにおいて出来るだけ分割数を少なくし、システムに掛かる負荷を小さくした。だが、3D モデルにおいて負荷の高いデータとなるのが頂点及び稜線であり、形状の変形において投影を用いるためひとつの形状につき 4 つの形状データを複製しているためモデルを構築してゆくうちに処理が遅くなってしまう。Dice では投影の消去や投影の表示・非表示の機能で対応したが、システム内のアルゴリズムの最適化やハードの性能の強化など抜本的な対処を行うことが出来れば処理速度の問題も解決すると期待できる。またそれに伴って分割数をケースに応じて変更し、形状モデルと投影モデルの位相要素を増やすことができれば、ユーザの意図通りであったり、より意匠的な形状を構築することが出来るだろう。

第 5 章

結論

本研究では従来のモデリングツールと比較して簡易なシステムでユーザがモデリングを、特にモデルの編集作業と配置、形状の 3 次元空間での位置やモデル全体の把握を容易にするインタフェースを提案した。従来のツールのインタフェースに捉われず、手書きジェスチャーでのストロークから形状を生成し、一面図や多面図でそれらを確認するのではなく、画面内に存在するキューブへ形状の頂点や稜線を上左右奥に投影することでモデルを俯瞰し、また投影の稜線や頂点をキューブの投影を参照しながら個別に操作する形態を取った。これによりモデリングの際にモデル全体の把握や配置、形状の変形を容易く行うことが出来た。前章で示したように問題点は多く、改善点は様々な部分で存在するが、ツールの完成度を高め手軽に使えるようなものに、3D モデリングという技術をもっと身近なものにしていきたい。

追記として、本ツール「Dice」の名称の由来を記す。Dice はキューブへの投影という独自のインタフェースを持つ。形状を四方の面に投影したキューブが正六面体のさいころ（賽、ダイス）を連想するためこの名称となった。

謝辞

本論分を締めくくるにあたり、終始温かく時には厳しくご指導をいただき、数え切れないほどの適切な助言、ご教授を頂きました渡辺大地講師に心よりの感謝の意を表します。

また、研究生活において大変お世話になりましたゲームサイエンスプロジェクト研究室の諸先輩方と学友に深く感謝します。

最後に、研究生活に潤いを与えてくれた東京工科大学吹奏楽団 TUTWinds、及び団員諸氏に心より感謝を申し上げます。

参考文献

[1] V.Branco, A.Costa, F.N.Ferriera, “Sketching 3D Models with 2D interaction devices”, Proc. of Eurographics, 1994.

[2] R.C.Zeleznic, K.P.Herndon, J.F.Hughes, “SKETCH:An Interface for Sketching 3D Scene”, Proc. of SIGGRAPH, 1996.

[3] M.Akeo, H.Hashimoto, T.Kobayashi, T,Shibusawa, “Computer Graphics System for reproducing three dimentional shape from idea sketch”, Eurographics ’94 Proceedings, 1994.

[4] 足立啓介, “デッサン初心者の習熟を想定したデッサン画からの 3 次元形状の生成”, 東京工科大学メディア学部メディア学部卒業論文, 1996.

[5] 三谷純, “意匠設計支援のための 3 次元スケッチに関する研究”, 東京大学大学院工学系研究科修士論文, 1999.

[6] 五十嵐健夫, 中嶋孝行, 小寺敏正, 田中英彦, “手書きスケッチによる自動車のボディ形状デザイン”, 1996.

[7] A.P.Pentland, “Local shading analysis”, IEEE Trans.Pattern Analysis and Machine Intelligence, 1984.

[8] 池内克志, “反射地図に基づき二次元濃淡画像から 3 次元形状を再構成する 2 手法”, 進学論, 1982.

[9] 鈴木敏博, 松田浩一, 近藤邦雄, “手書きスケッチにおける陰影表現を用いた 3 次元形状制御手法”, 情報処理学会論文誌「グラフィックスと CAD」, 2002.

[10] S.Sugishita, K.Kondo, H.Sato, S.Shimada, “Sketch Interpreter for geometric modeling”, Annlals of Numerical Mathematics 3, 1996.

[11] 古島終作, 金井理, 高橋秀智, “手書き図形の自動認識による 3 次元自由曲線モデルの生成”, 精密機械工学会誌, 1993.

[12] T.Tanaka, S.Naito, T.Takahashi, “Generalized symmetry and its application to 3D shape generation”, Visual Computer, 1989.

[13] L.Markosian, M.A.Kowalski, S.J.Trychin, L.Bourdev, D.Goldstein, J.F.Hughes, “RealTime Nonphotorealistic Rendering”, Proc. if SIGGRAPH, 1997.

[14] D.Pugh, “Designing solid objects using interactive sketch interpretation”, Computer Graphics, 1992.

[15] T.Igarashi, S.Matsuoka, H.Tanaka, “Teddy:A Sketching Interface for 3D Freeform Design”, Proc. of SIGGRAPH, 1999.

[16] TAITO, ラクガキ王国公式サイト, <http://www.garakuta-studio.com/>.

[17] e-frontier, マジカルスケッチ 2, <http://shade.e-frontier.co.jp/magical/>.

[18] 渡辺大地, FK Tool Kit System, <http://www.media.teu.ac.jp/~earth/FK/>.

[19] 魏大名, 先田和弘, Roman Durikovic, 向井信彦, Carl Vilbrandt, 「コンピュータグラフィックス」, オーム社, 2003.

[20] 平岡和幸, 堀玄, 「プログラミングのための線形代数」, オーム社, 2004.