平成 11 年度修士論文

意匠設計支援のための
3 次元スケッチに関する研究

指導教官 鈴木宏正 助教授

東京大学大学院工学系研究科
情報工学専攻

86883 三谷 純

概要

近年では工業製品の設計過程において、 CAD システムを用いることで作業の効率化を図ることが一般的となってきている。しかし、 CAD を用いた設計作業は一般に操作が繁雑で形状の構築を直感的に行うことができないため、試行錯誤を伴う意匠デザインの段階では、旧来からの 2 次元の紙面上へのスケッチにとどまることが多い。またアイデアの創出段階において、 CG による無機質な 3 次元形状の表示は発想の展開の妨げになることがあり、スケッチ図のより豊かな表現力が見直されてきている。

そこで、デザイナが従来のスケッチと同等のインターフェースで 3 次元モデルを構築する事ができ、さらにデザイナの描画スタイルを活かしたモデルの表示が可能であれば、従来のスケッチの優れた点と、計算機による 3 次元化の利点を活かすことができ、意匠デザインの効率化と発想の支援の両面で役立てられると考えられる。このような新しい手法を 3 **次元スケッチ**と呼ぶこととし、本研究では意匠設計段階における発想支援と 3 次元モデル構築の支援を目的とした、 3 次元スケッチの手法を提案する。

目次

図目次

第 1 章

序論

第 1 章

序論

1.1　はじめに

近年では工業製品の設計生産過程において、計算機を利用してその自動化や能率の向上を図る CAD/CAM(Computer Aided Design/ Computer Aided Manufacture) システムが一般的なものとなってきている。これにより、設計の上流から下流への一貫したデジタルデータのやりとりで、作業の効率化による開発期間短縮が図られるようになってきている。

しかしながら CAD を用いた設計作業は、複雑なメニューやコマンドを使用して厳密な形状を定義する必要があり、時間と労力がかかる作業である。そのため、試行錯誤を伴う意匠デザインの段階は、いまだデジタル化による設計支援が困難な分野であり、旧来からの 2 次元の紙面上へのスケッチにとどまることが多い。

ところで、人間が頭の中でイメージしている形状は曖昧である場合が多く、実際にスケッチを行いながら、徐々に具体化していくことがよく行われる。この場合、 2 次元のスケッチは 3 次元形状を表現するためだけのツールとしてではなく、人間の発想を支援するためのツールとしても使用されているといえる。意匠デザインの段階では、いかにしてアイデアを創出するかが重要であるため、計算機によって発想を支援することが可能であれば、非常に有益であると考えられる。

また一方で、計算機内に構築された 3 次元モデルの出力には高度な光学計算を用いた CG 技術を用いることが一般的であったが、敢えて輪郭を曖昧に表現したり、特徴的な形状部分を強調表示するなどによって、デザイナの望むような表示を行う、ノンフォトリアリスティックレンダリングについても研究されるようになってきている。

そこで、デザイナが従来のスケッチと同等のインターフェースで形状の描画を行う事によって 3 次元モデルを構築する事ができ、さらにデザイナの描画スタイルを活かしたモデルの表示が可能であれば、様々な角度から眺めたスケッチ図を直ちに得る事ができ、意匠デザインの効率化と発想の支援の両面で役立てられると考えられる。

本研究では、このようにデザイナが従来のスケッチと同等のインターフェースで 3 次

元モデルを構築し、スケッチスタイルを活かしたモデル表示を行う手法を 3 **次元スケッチ**と呼ぶこととし、意匠デザイン段階における発想とモデル構築の支援を目的とした 3 次元スケッチシステムを提案する。

1.2 意匠設計段階での計算機による設計支援の可能性

近年では製品の多様化とライフサイクルの短期化が進み、意匠設計においても設計の効率化が求められるようになってきている。ここでは、近年の意匠設計段階におけるデザイナをとりまく環境の変化をまとめ、アイデア発想段階におけるスケッチの役割と、その特徴について考察する。次に、従来の CAD を意匠設計段階に用いる際の問題点を明確にし、計算機によって可能な設計支援方法についてまとめることで、本研究で提案する 3 次元スケッチが置くべき主眼を述べる。

1.2.1 デザイナをとりまく環境の変化

最近の意匠設計に関する調査は、昨年度にまとめられた「コンテンツ制作支援システムに関する調査研究報告書 [機械システム振興協会 99]」に詳しい。ここでは、この報告書をもとに、デザイナをとりまく環境の変化をまとめる。

３次元 CAD の普及と生産効率の追求

近年では意匠設計の後工程である設計部門や製造部門から、3 次元データによるデザインの提出が求められるようになってきている。大企業を中心に設計・製造部門への 3 次元 CAD の導入が急速に進み、設計・製造部門では、同一の 3 次元モデルデータを各プロセスで共有することが可能となった。これにより、大幅な生産性の向上に成功した。これをデザイン部門にまで敷延しようという動きが現在盛んである。

デザイン部門から提出された 3 次元 CAD データを元に機構部品を組み込んだり、金型データを作成できれば、生産効率は向上する。また、場合によっては機構部品を大まかに配置した 3 次元 CAD データをデザイン部門に提出し、それをもとに意匠設計を行うことも考えられる。この場合、デザイン部門にとっての 3 次元 CAD は後工程である設計・製造部門とのインターフェースという意味合いを越え、デザイン作業そのものを行う基本的な作業ツールと位置づけられる。この場合、設計で用いられているものと同等の機能を備えた CAD システムがデザイン部門でも要求されることとなる。

デザインの幅を広める

上記の、生産性向上のための 3 次元 CAD/CG ツールの導入とは別に、デザインを発想するときの支援ツールとして計算機を使用することも検討されるようになってきている。

例えば、特定の形状からさまざまなバリエーションを派生させたり、計算機内でアニメーションを作成することなどにより、デザイナの発想を支援することが実際に行われている。つまり、従来手作業では困難であった作業を、仮想現実の世界でシミュレートすることにより、デザイナ本来の活動の幅を広げようとするねらいである。この場合、従来の CAD とは異なり、迅速かつインタラクティブにモデルの構築と表示を行える機能を備えたシステムが要求される。

1.2.2 発想段階におけるスケッチの役割

意匠設計におけるスケッチには、二つの役割があり、それぞれ異なる方法で描かれる。一つは、デザイナが自分のアイデアを確認したり、さらに展開させるために行うスケッチで、サムネイルスケッチ (図 1.1)、メモスケッチ (図 1.2) などとよばれる、第三者に伝達する必要のないものと、もう一つはデザイナが自分のデザインの意図を第三者に伝達し理解を求めるために描かれるスケッチで、プレゼンテーションスケッチなどと呼ばれているものである [清水吉治 98]。

サムネイルスケッチ、メモスケッチ

サムネイルスケッチは親指程度の大きさに描く小さなスケッチ、メモスケッチはイメージしたものを記録するための小さなスケッチをいう (図 1.1)。両者とも、デザインプロセス初期の段階に、デザイナが自分のイメージを展開や確認をしたり、デザインの基本構成を大まかにまとめたりすることを目的に描かれるスケッチで、それを描いた本人が理解できればよいわけであるから、第三者に見せて伝える性質のものではないといえる。

プレゼンテーションスケッチ

コンセプトに基づき、デザインが自分のイメージを展開、発展させまとめたものを第三者に伝達し、理解を得たり、それぞれのデザインを比較検討するために描かれるスケッチで、おそらくプロダクトデザイナの描画頻度が最も高いスケッチといえる (図 1.2)。

1.2.3 デザイナによるスケッチの特徴

「コンテンツ制作支援システムに関する調査研究報告書 [機械システム振興協会 99]」に付随する、デザイナによるスケッチの様子を収録したビデオ (約 6 時間) を視聴することで、実際にデザインがスケッチを行う際の特徴を観察した。その際、スケッチ時に見られた特徴を以下にまとめる。なお、ビデオの内容は、特にサムネイルスケッチ、プレゼンテーションスケッチの区別無く、プリミティブ形状のスケッチから、既存の製品を眺めながらのスケッチなどを行ったものを、撮影したものである。

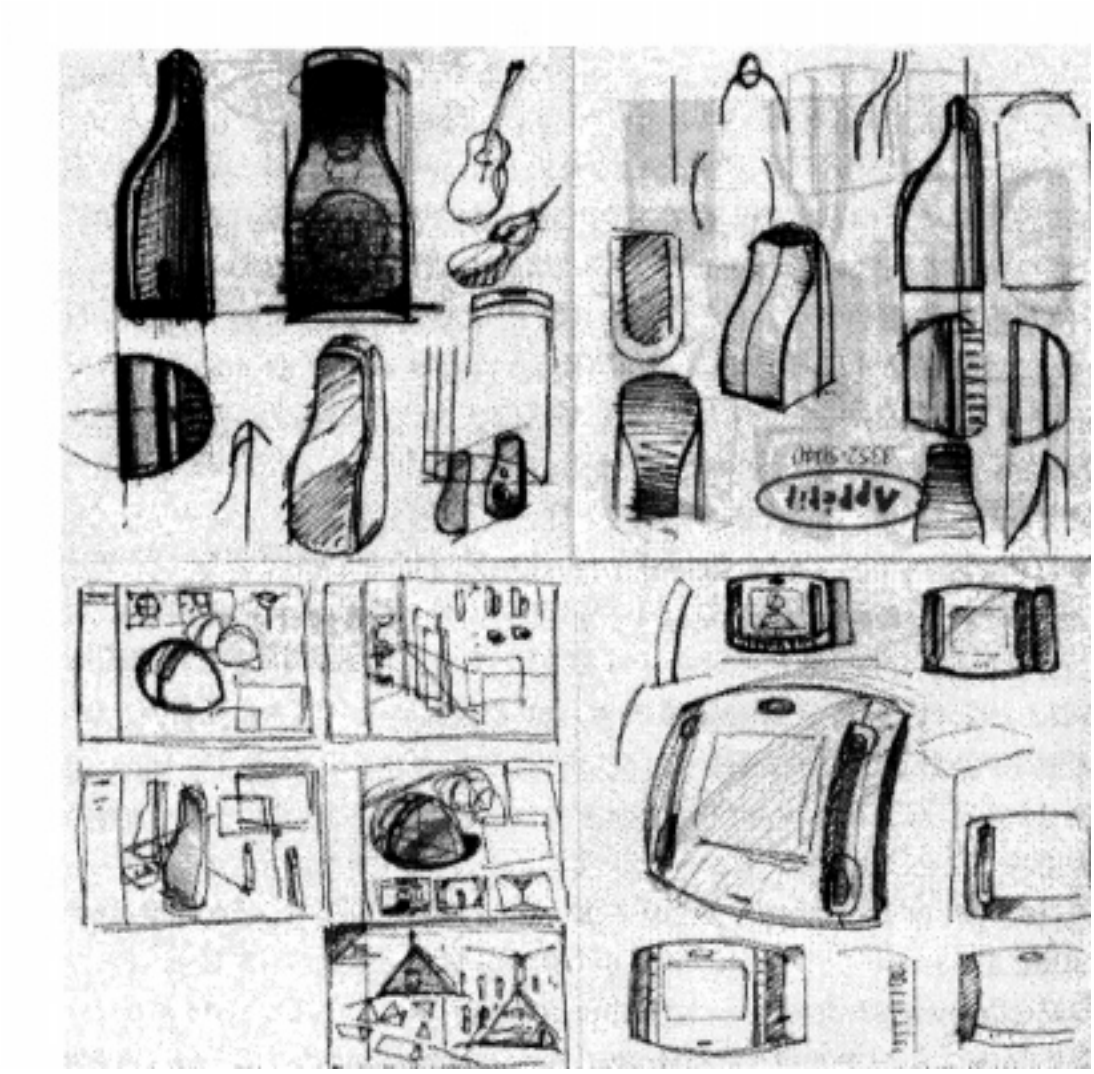


図 1.1: サムネイルスケッチ
[機械システム振興協会 99]

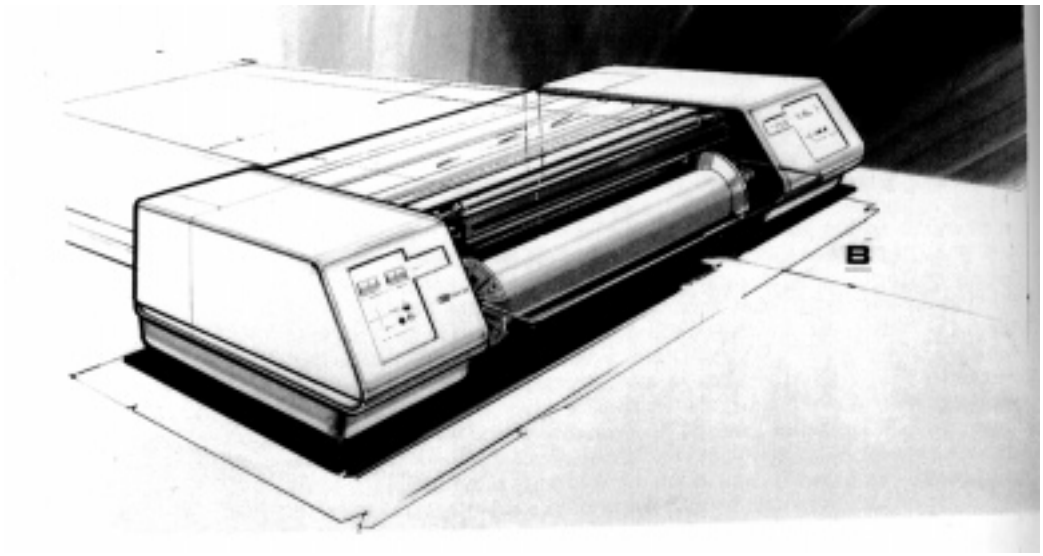


図 1.2: プレゼンテーションスケッチ
[清水吉治 98]

- 描画スタイル
  - 紙を回転させて描く
  - 書きやすい角度、書きやすい向きがある (個人の好み・人間工学的問題)
  - 球を書く際に、何度も輪郭をなぞるように描き、徐々に形を整える
  - あまり細切れの線は書かない (連続した線の流れを重視する)
  - 線の勢いを活かす
  - パースをかなり意識し、正しい投影図を描くようにする
  - 定規を使うことがある
  - パースを決めるために、線を長めに書く
  - 最初に書いた薄い線を補助線として細部を書いていく
- 線の太さ
  - 線を太くしながら形状を補正する
  - 線の太さに意味を持たせる (強調する箇所を太く描く)
  - 同じ線を何度もなぞって太くする
  - 手前に見える部分を強く書く
- 陰影

  - 3 次元形状らしく見せるために陰影をつける

- 断面線
  - 外形から 3 次元形状が推測できる場合は断面線は書かない
  - わかりにくい場合に、説明のために断面線を書く
- 修正
  - 消しゴムは使わない
  - 修正は、線の重ね書き、もしくは始めから書く
- CAD について (デザイナの言葉から)
  - CAD を使うことで早く立体的に検証できること
  - コンピュータ上の制約に思考がひっぱられることがある
  - 曖昧な感じを表現できない
  - CAD の操作に詳しいと、 CAD のモデリング方法とスケッチ方法が似てくる
- その他
  - デザイナが CAD でデザインしたとしても、それをそのまま金型設計に使えるわけではない
  - 断面線を入力してモデルを構築することはあまりやらない
  - レンダリングの方法で 3 次元形状の見え方が違う
  - スケッチにはレンダリング結果を自分で決められるメリットがある
  - 人によって物体を描く順番が違う
  - 一枚の絵で形状が把握しにくい場合は複数枚のスケッチを描く

1.2.4 スケッチにおける形状構築段階と形状表現段階

スケッチには、 2 次元の紙の上に透視図を描くことで立体形状を構築していく段階 (図 1.3上) と、構築された投影図をより立体的に質感豊かに見せるために表面を描いていく段階 (図 1.3下) がある。ここでは、それぞれを「スケッチにおける形状構築段階」と「スケッチにおける形状表現段階」と呼ぶことにする。これは、 CAD や CG ソフトでの「モデリング」と「レンダリング」の操作に対応するものだと考えられる。

つまり、スケッチにおいても計算機を用いた場合においても、まずは形状を構築するステージがあり、その後で、その形状をより表現豊かに見せる表示のためのステージがあるといえる。

それぞれのステージにおいて、 CAD システムに対するスケッチ描画のメリットは、以下のようにまとめられる。

- 形状構築段階

  スケッチでは描画する形をペンの動きで表現できるため、形状の構築が直感的であり、また具体的に決まっていない箇所については曖昧な形状を残したまま形状の作成を進められる。既存の CAD システムでは、メニューの選択やコマンドの入力、数値の指定が、形を作ることと直感的に結びつかないため、意匠デザインの段階では、これらの操作が煩わしく、アイデア創出の妨げとなる。また曖昧さを残したまま形状を構築できない。

- 形状表示段階

  スケッチでは強調したい箇所を太く書いたり、形状を把握しやすいようにデフォルメしたり、陰影をつけたり、デザイナの好みで表現方法を工夫することができる。既存のレンダリングシステムではデザイナの意図した表現が行えず、 CG 独特の無機質な表示となりがちである。

1.3 本研究の目的

スケッチには、これまでにまとめたような、従来の計算機を用いたシステムでは実現が困難である優れた点が多くある。そのため、計算機によって意匠デザインを支援する際には、これらスケッチを行う事のメリットを奪わないように注意を払う必要がある。

また、スケッチにおいても形状の構築段階と形状の表現段階という異なるステージがあるため、それぞれの段階における現状の CAD システムの問題を意識する必要がある。ただ、この 2 つの段階は明確に区別できるものではなく、スケッチ作業においては、両方を行き来することも多い。

そこで本研究では、形状の構築段階においては、スケッチと同等のインターフェースによって 3 次元形状を構築し、形状の表現段階においては、スケッチで使用される表現手法と同じような表示方法を行うことが可能な、新しい 3 次元スケッチシステムを提案する。なお、このような表示手法をこれ以降**手描きレンダリング**と呼ぶこととする。このシステムの主眼は次の 2 点にまとめられる。

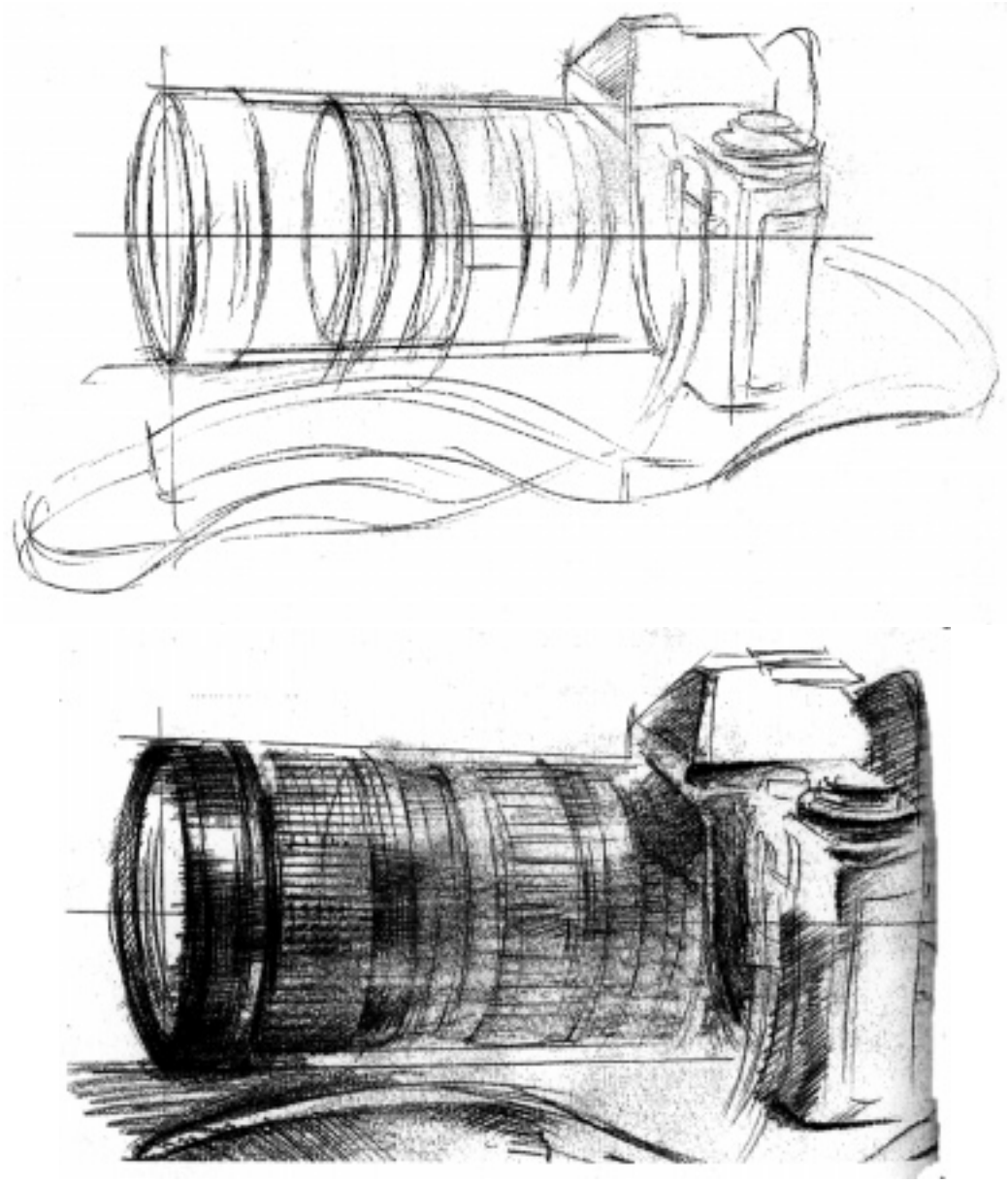


図 1.3: スケッチにおける形状構築段階 (上) と形状表現段階 (下)
[視覚デザイン研究所編集室 94]

- 3 次元形状モデリングを意識することなく、紙の上に行なっているスケッチと同等の入力から 3 次元形状を構築する
- デザイナの描画スタイルを活かした 3 次元形状の表示を行う

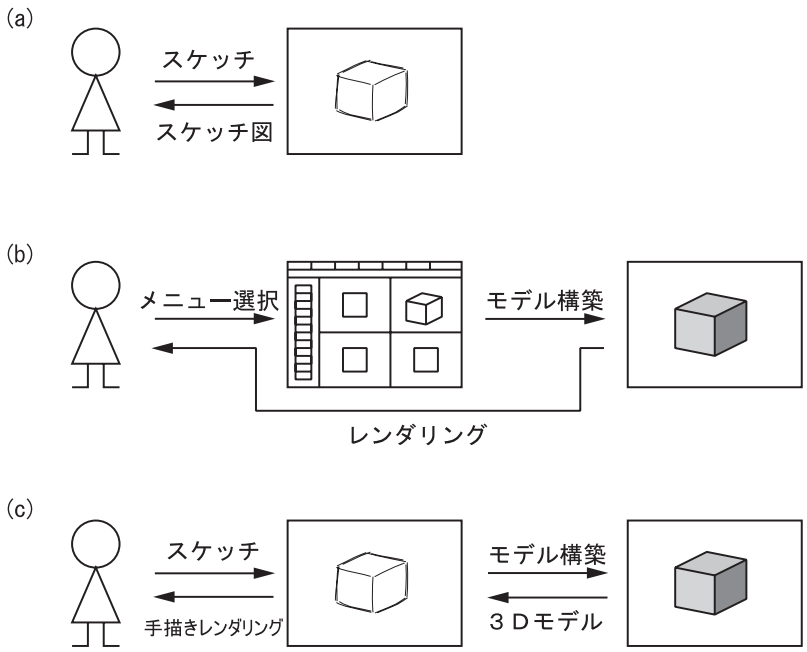


図 1.4: スケッチと 3 次元 CAD および 3 次元スケッチのインターフェース

実際にデザイナが行っている紙面上でのスケッチと、従来の 3 次元 CAD、および本研究で提案する 3 次元スケッチのインターフェースを図 1.4に示す。紙面上でのスケッチ (図 1.4(a)) では、デザイナは入力としてスケッチを描き、そのスケッチ図を出力として視覚的に確認することを行う。一方、従来の 3 次元 CAD(図 1.4(b)) では、デザイナはメニューを選択したり値を入力することで 3 次元形状を構築し、そのレンダリング結果を得る。本研究で提案する 3 次元スケッチ (図 1.4(c)) では、デザイナが入力したスケッチをもとに、システムが自動的に 3 次元形状の構築を行い、その 3 次元形状を手描きレンダリングで出力を行う。

このシステムが実現すれば、従来のスケッチと同じインターフェースで 3 次元モデルを構築できるようになり、構築したモデルを別の角度から眺め、形状の検討を行うことができる。また、別の角度から眺めたものでありながら、スケッチされたものと同じようなスタイルで形状が表示される。入力がスケッチ図であるため、特に 3 次元モデリングを意識することなく、デザイナには負荷がかからない。また、得られた形状データを別のアプリケーションで利用する事も可能となる。

ところで、試行錯誤という面からは、入力スケッチ図をバッチ処理 (一括処理) するのではなく、対話的にモデルを構築してゆくことが望まれるであろうが、そのように

するには、常に 2 次元と 3 次元を行き来できるようにする必要があり、一入力ごとに 3 次元形状を構築しなくてはならない。この場合、複数の入力によって初めて 3 次元形状が定まるような形状の入力ができない問題点がある。

そのため、ある程度の形状をスケッチ図から入力し、バッチ処理で 3 次元形状を構築してしまった後、得られた形状に対する修正、加筆などを対話的に行うことができればよいと考えられる。

本研究では、スケッチ入力から 3 次元モデルを構築し、その形状をスケッチ入力されたものと同じように表示するまでの手法を提案する。

1.4 本論文の構成

本論文は全部で 5 章からなる。

2 章では、計算機内に 3 次元形状データを入力するための手法と、それに関連する研究、および計算機を用いた 3 次元形状の表現手法にについてまとめる。

3 章では、本研究で提案する新しい 3 次元スケッチシステムについて述べる。スケッチと同等のインターフェースによる 3 次元モデルの構築手法と、デザイナの描画スタイルを活かした手描きレンダリングの手法について具体的なアルゴリズムを述べる。

4 章では、本研究で提案した 3 次元スケッチシステムを適用した結果とその評価をまとめる。

最後に、 5 章で本論文の結論と今後の展望について述べる。

第 2 章

計算機を用いた 3 次元形状モデリングとモデル表示手法

第 2 章

計算機を用いた 3 次元形状モデリングとモデル表示手法

近年では計算機を用いて 3 次元形状を扱うことは極めて一般的なこととなっている。しかし、計算機への形状の入力は、いまだ時間と労力のかかる作業であるため、そのコストを軽減するための新しい形状入力手法については、研究の余地が多く残されている。また、計算機によって形状の画像を出力する手法については、既に実写と見まがうほどリアルな出力を得られるようになっている。近年ではより形状の把握に適した表現手法として、アニメ風や手描き風に表示するための研究が盛んに行われている。本章では、既存のモデル構築手法とスケッチ入力による 3 次元モデル構築のための関連研究、およびモデルの表示手法についてまとめる。

2.1　計算機を用いた 3 次元形状モデリング

3 次元形状モデルを計算機内に構築するための手法は古くから研究され、数多くの手法が存在する。工業製品の設計分野においては WIMP(Window Icon Menu Pointing device) インターフェースによる CAD を用いたモデリングが一般的であるが、これは時間と労力がかかる作業であるため、近年では 2 次元の画像を元に 3 次元形状を再構築する手法や、より直観的なインターフェースでモデルを構築するための研究が多くなされている。また、実在する 3 次元形状モデルを測定したデータを入力とすることで、3 次元形状を計算機内に構築することも一般的に行われるようになってきている。ここでは、計算機を用いた 3 次元形状モデリングの手法について、モデルの構築に必要なデータと、その入力形態に応じて図 2.1のように大まかな分類を行ってみた。

まず、既存の手法を大きく二つに分けると、入力として直接 3 次元データを取得するものと、入力されるデータが 2 次元であり、それを元に 3 次元形状を構築してゆくものに分けることができる。

3 次元データを入力とするものには、実在モデルをデジタイザや非接触型の 3 次元スキャナーなどで計測することによって 3 次元データを取得するもの (図 2.1(a)) と、データグローブなどの 3 次元情報を直接入力できるデバイスを用いて対話的に形状を入力

図 2.1: 入力形態による 3 次元形状モデリング手法の分類

するものとがある。後者には、得られた 3 次元情報を直接形状に反映するもの (図 2.1(b)) と、 3 次元情報をジェスチャーとして捉え、それを元に特定の形状を構築するもの (図 2.1(c)) がある。

他方、 2 次元のデータを入力とするものには、スケッチ図をスキャナで読み込み、バッチ処理するもの (図 2.1(d)) と、対話的にモデルを構築してゆくものに分類できる。

対話的な処理を行うものの代表的な例として、 WIMP インターフェースによる既存の CAD システム (図 2.1(e)) がある。また、近年研究がなされている手法として、ジェスチャーを用いたもの (図 2.1(f)) や、 2 次元で入力されたストロークを直接 3 次元形状に反映させる手法 (図 2.1(g)) がある。

次節以降、それぞれの手法について具体例を紹介する。

2.1.1 実在モデルの計測データを用いたモデル構築

実在する 3 次元物体を計測して 3 次元モデルを計算機内に構築する手法は、実際に手に取って知覚できるモデルを計測対象とすることで、形状を正確に把握できるという利点がある。測定によって得られる情報は一般に、 3 次元モデルの表面上の点群座標であるため、得られた点群から正しい位相を持ったモデルを再構築する必要があり、そのための研究が多くなされている [Curless96][Eck96]。アニメーションキャラクターなど、既存の CAD や CG のシステムでは形状の入力が困難な有機的な形状に対して、有効な手法である。この場合、一度クレイモデルを作成し、それを非接触型の 3 次元スキャナーなどで計測することによってモデルの入力を行う。研究の盛んな分野であるが、本研究では計算機内で 3 次元モデルを構築することに主眼をおくため、ここでは簡単な紹介にとどめる。

2.1.2 3 次元デバイスを使用したモデル構築

3 次元デバイスを用いることで、立体情報を持つデータを入力として扱えるため、3 次元モデルを直接構築することが可能である。

この手法の 1 つに、ユーザの手の動きに応じて 3 次元空間内に曲面を生成し、モデルの構築を行う Surface Drawing がある [Schkolne99]。ユーザは位置センサー付きのデータグローブをはめ、立体視用の HMD(ヘッドマウントディスプレイ) を身につけることによって、仮想現実の世界で 3 次元モデルを構築する (図 2.2)。生成される面は、指によって定義される頂点群と法線ベクトルを元に計算される (図 2.3)。この研究では、消しゴム機能や面を滑らかにする機能も実装されており、親指の位置でそのオンオフの切り替えを行っている。

また、3 次元デジタイザを 3 次元のペンに見立て、3 次元のストロークを入力としてする手法もある [Han97]。この場合、入力は面情報を持たない曲線となるため、特定の曲線入力に付加的な意味を持たせて 3 次元オブジェクトの生成を行っている (図 2.4)。

また、3 次元空間における腕のジェスチャーによってプリミティブの生成を行う研究もなされている [西野 99](図 2.5)。生成したプリミティブに対するねじりや丸め操作など、生成後の形状操作もジェスチャーで行えるようになっている。

3 次元のデータ入力を用いる手法は、2 次元で行っているスケッチ作業をそのまま 3 次元に拡張した点で、直感的な作業をダイレクトに形状に反映することが可能である。また、クレイモデルを実際に作成する場合に比べ、大きなモデルを作成する負荷が小さく、コストもかからない。しかし、生成した 3 次元モデルに触れることができないため、形状の変形や詳細部の作り込みが困難であり、また HMD やデータグローブのような大がかりで高価な装置が必要であるという問題がある。

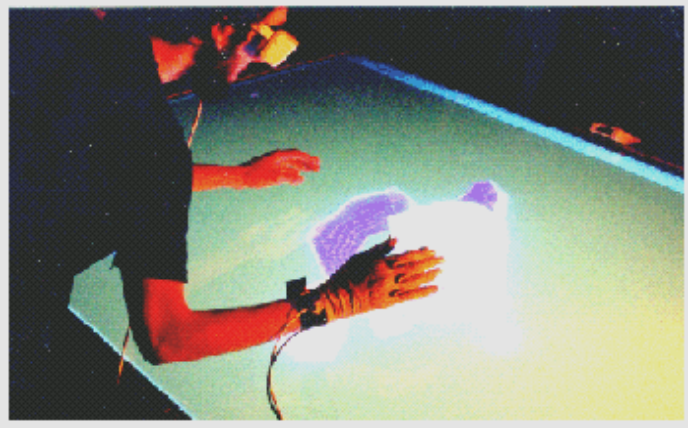


図 2.2: Surface Drawing のインターフェース
[Schkolne99]

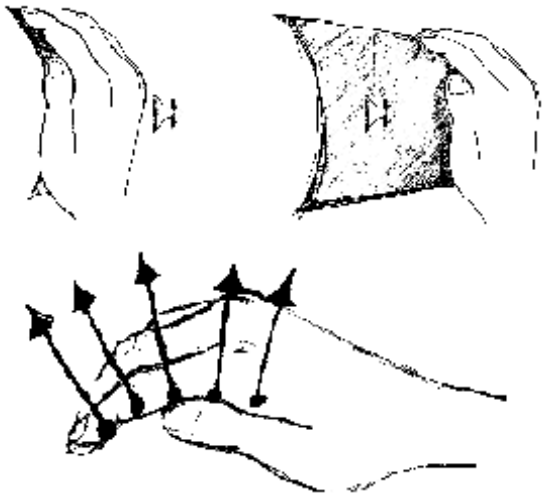


図 2.3: Surface Drawing での面の生成
[Schkolne99]

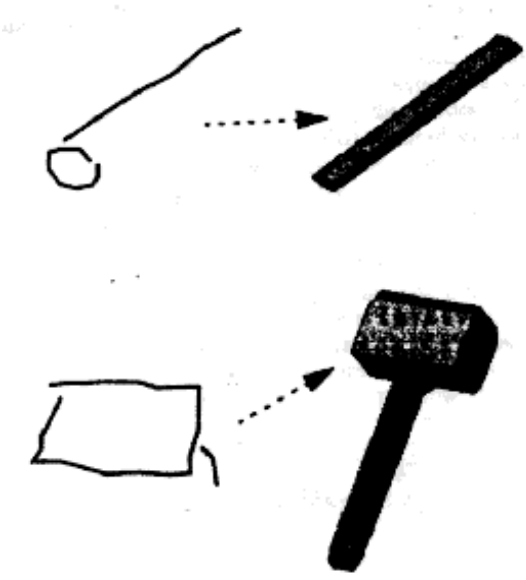


図 2.4: 3 次元空間内でのストロークによるモデル構築
[Han97]

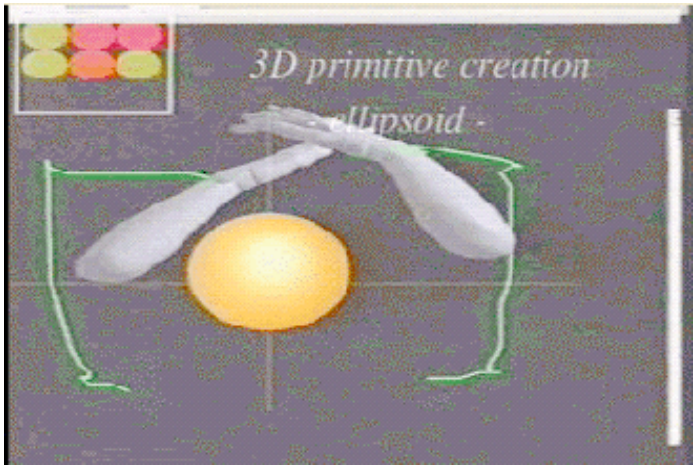


図 2.5: 腕によるジェスチャーでのモデル構築

[西野 99]

2.1.3 スケッチ図を利用したモデル構築

デザイナが描いたスケッチ図をスキャナなどで計算機に読み込み、それを元に 3 次元形状を構築しようとするアプローチは広く行われている。

[Akeo94] は、実際にスケッチ図をスキャナで読み込んだ後、それを元に 3 次元形状の復元を行った。一般に一枚の 2 次元画像から 3 次元形状モデルを一意に復元することは不可能であるため、色分けされた断面線を追加することで不足する情報を補っている。 3 点透視で描かれたスケッチ図に対しては、 3 つの消失点からデザイナの視点を求めることが可能で [Sugishita96]、追加された断面線の情報から各頂点の 3 次元座標を計算している (図 2.6)。

また、 Design Spinnaker[(株) エヌケー・エクサ] では、スケッチ図の中の形状特徴線をユーザがディスプレイ上でなぞることによって 2 次元曲線を入力し、その後で対称性などの拘束条件から、 3 次元の曲線と曲面の構築を行っている。

これらの手法は、今まで CAD のオペレータが経験と勘に頼って行っていた、 2 次元スケッチから 3 次元形状を再構築する作業を支援することを目的としている。デザイナは今まで通り紙の上でスケッチをするため、負荷なく使用できる。現在は特徴線などの曲線情報しか抽出しないものが多いため、影や表面線のような、特徴線以外にも入力されている様々な情報を利用できていない。

2.1.4 2 次元入力の対話的処理 (ジェスチャーによる入力)

特定のマウスの動きと、それによって生成される形状の対応を予め規定しておく事で、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が [C.Zeleznik96]

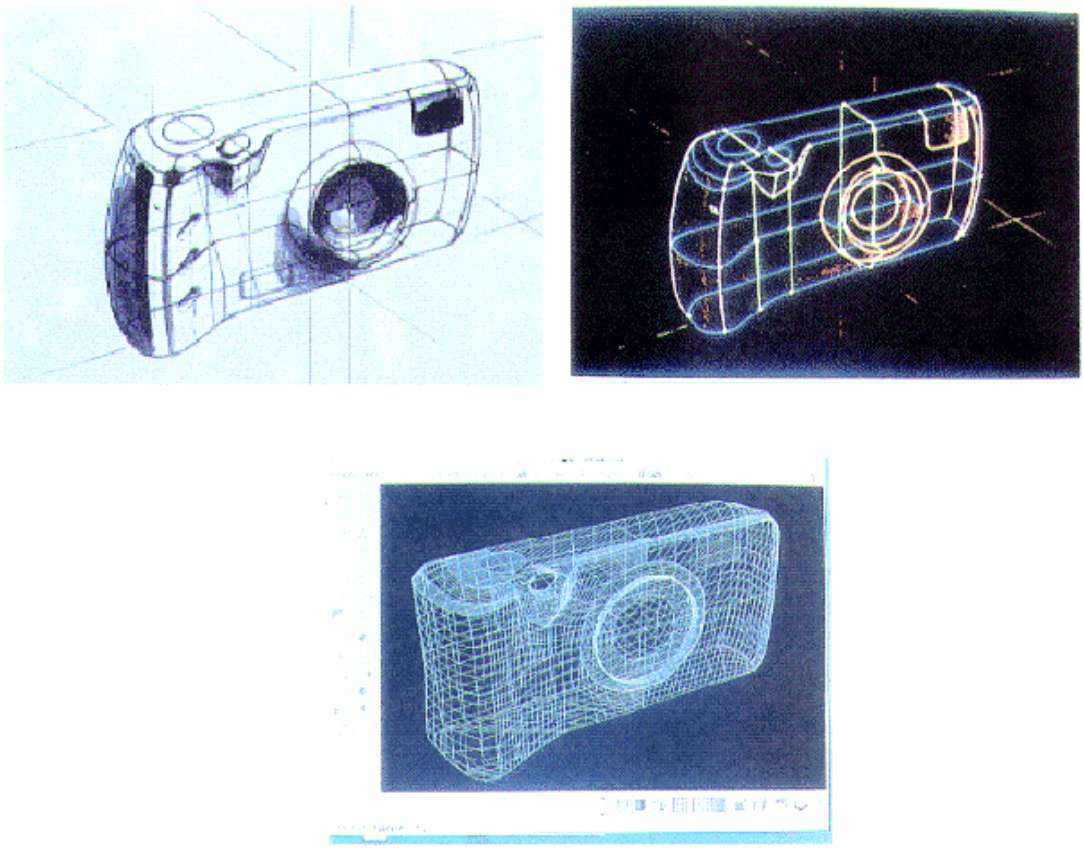


図 2.6: 断面線を追加したスケッチ図からの 3 次元モデル構築
[Akeo94]

と [Branco94] によって研究されている。ジェスチャーと、ジェスチャーによって生成される形状の対応を、スケッチを行う時のペンの動きと、スケッチによって表現される形状の対応に近付けることで、3 次元のプリミティブ形状を直感的に、かつ迅速に入力できるようになっている。特に [C.Zeleznik96] の SKETCH システムでは、スケッチの描画スタイルに近い直感的なジェスチャー群が多数用意され、簡単なインターフェースで多くのプリミティブ形状を生成することができる (図 2.7)。また、生成されたプリミティブ形状が 3 次元空間内のどこに配置されるかを 2 次元の情報からのみ決定する必要があるが、生成された物体と既存の物体との位置関係に拘束を与えたり、影の入力による追加情報を用いることで、違和感のない 3 次元配置を実現している。また、ブール演算やコピー、スケール変換など、一般的な CAD で行う操作がマウスのストローク入力で行うことができ、非常に完成度の高いシステムとなっている。

ただし、このシステムでは自由曲面を持つ形状を入力することができないという問題がある。また、ジェスチャーによる入力は、あくまで CAD システムに見られる多くのメニューやコマンドをストローク入力に置き換えただけにとどまりがちで、規定された形状しか入力できないため、実際のスケッチのようにユーザーのストロークを直接反映した形状の入力を行うことが難しい。

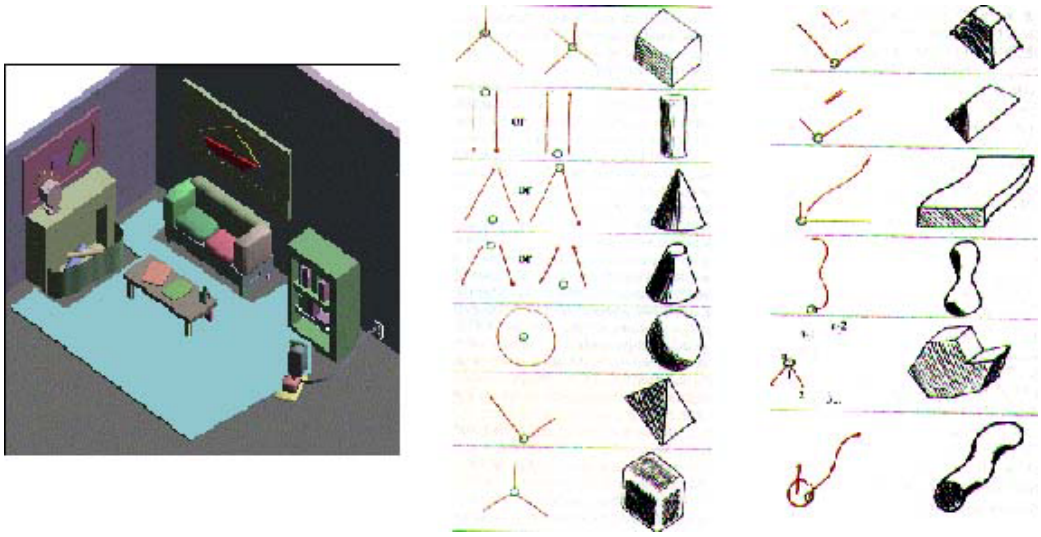


図 2.7: SKETCH システムで構築されたシーン (左)、ジェスチャーと生成されるモデルの対応 (右)

[C.Zeleznik96]

2.1.5 2 次元入力の対話的処理 (ストロークの直接利用)

計算機に 2 次元のストロークを入力し、そのストロークを直接入力形状の一部として使用する手法も研究されている。 [古島 93] は、ユーザが計算機上に描画した 2 次元曲線に対し、形状が対称であるという拘束条件を与えることで、 3 次元曲線を生成する手法を提案した。また、対称でない形状については、異なる視点からの線画入力によって 3 次元座標を求めている (図 2.8)。この論文の著者の一人によってフリーウェア「六角大王 [古島]」が作成され、多くのユーザに支持されている。

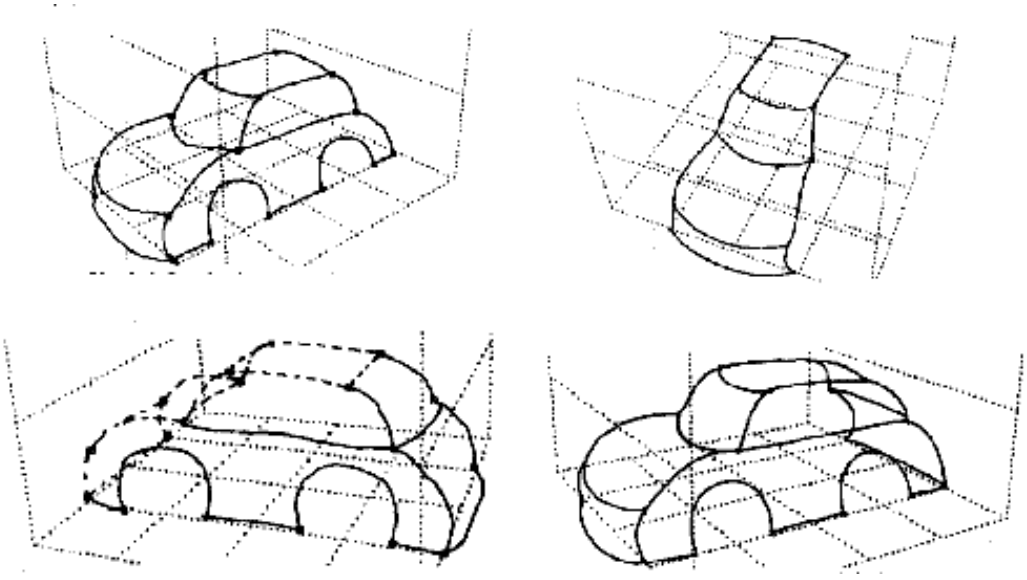


図 2.8: 対称性、異なる視点からの情報による 3 次元曲線の構築
[古島 93]

[Igarashi99] は、計算機に入力された 2 次元の閉じた曲線から、奥行きをの値を経験則から求めることで、球と同位相の 3 次元形状を生成する手法を提案した。さらに、ストロークによる形状操作機能を追加する事で、ぬいぐるみのような自由曲面モデルをスケッチ感覚で容易に構築できるシステムを作成した (図 2.9)。

いずれも、デザイナによって入力されたストロークの形状が直接 3 次元形状に反映されている点において、他の手法よりも優れていると考えられる。

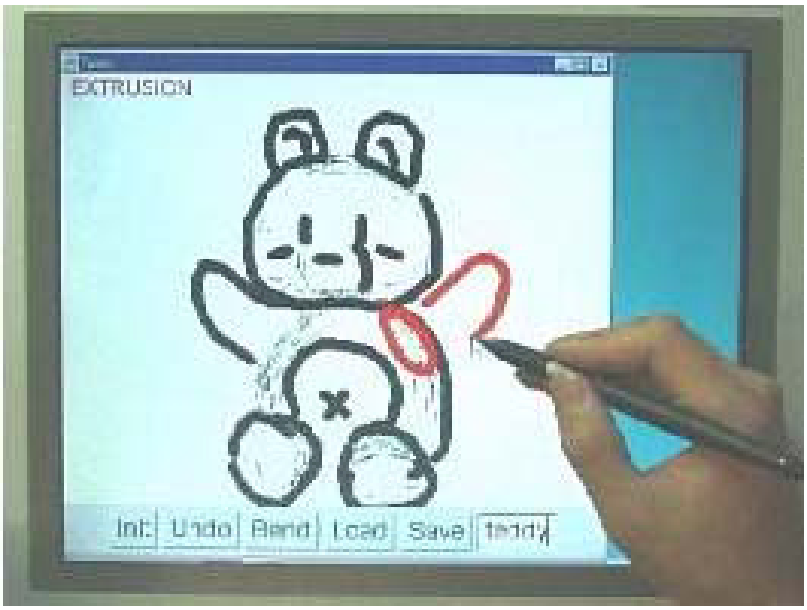


図 2.9: スケッチ入力からのぬいぐるみモデル作成例
[Igarashi99]

2.2 スケッチ入力による 3 次元モデル構築のための関連研究

前章では、 3 次元形状モデルを計算機内に構築するための様々な手法を紹介したが、その中でもデザイナにとって好ましい手法は、 2 次元のスケッチ入力から 3 次元形状を構築する手法であろう。ここでは、 2 次元のデータから 3 次元モデルを構築するためのアルゴリズム、およびそれに関連する手法をいくつか紹介する。

2.2.1 複数の視点からの投影図を用いた座標値の決定

視点の位置が既知である場合、 2 つの異なる視点からの情報を用いることで、 3 次元座標値を計算によって求めることが可能である [古島 93]。また視点の位置が未知である場合であっても、 [Debevec96] は 6 枚以上の写真から建築物を再現する効率的なアルゴリズムを提案している。しかし、実際に手描きスケッチによって複数の視点からのスケッチを入力した場合には、データに誤差が含まれ、場合によっては復元不可能な入力も有り得るため、スケッチ入力に対応するには柔軟なシステムが必要となる。また、この手法で構築される形状は、異なる視点間で対応をとることが可能な点や線分に限られるため、球体のような、滑らかな曲面に囲まれた形状の復元は困難である。

2.2.2 対称形状である拘束条件を用いた座標値の決定

入力形状が特定の平面に関して対称形状であることがわかっている場合、互いに対称である点の情報が、異なる 2 つの視点からの入力と同等に扱うことができるため、3 次元座標値の復元が可能である。1 枚の入力から 3 次元の座標値を求めることができるため、スケッチと同等なインターフェースで 3 次元モデルを構築するには非常に有効な手法である。本研究で提案する 3 次元スケッチにおいても、この手法を一部で用いている。アルゴリズムの詳細は 3.5.5節に記す。また、我々が日常目にする物体について、左右対称なものは多く存在するので、この手法を利用できる場合も多い。全体として対称形状ではなくても、個々のパーツが対称形状であるものを組み合わせて複雑な形状を構築することも可能である [Tanaka89](図 2.10)。

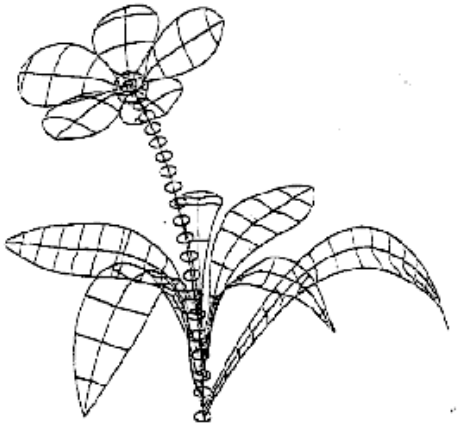


図 2.10: 対称形状と円柱形状によって作成されたモデル
[Tanaka89]

2.2.3 3 点透視法による視点推定

透視図が 3 点透視法で描かれている場合、3 つの消失点から視点を推測することができ、それを元に各頂点の 3 次元座標を求めることができる。この手法については [Sugishita96] に詳しい。本研究では、3 つの消失点が求まっていない場合でも、視点の位置について仮定を設けることで視点位置を推定する手法を 3.5.1節にて提案し、3 次元スケッチシステムに使用している。

2.2.4 幾何推論による位相決定

稜線が図 2.11(左) のように描かれている場合、頂点辞書 (図 2.11(右)) を各頂点に当てはめることにより、稜線の凹凸を判定することができる。これをすべての稜線に対

して行うことで、形状の位相を決定することができる [P.H. ウィンストン 79]。しかし、矛盾の無い厳密な入力が必要であるため、スケッチ入力や、画像処理によって得られた誤差を含む形状を扱えない問題がある。

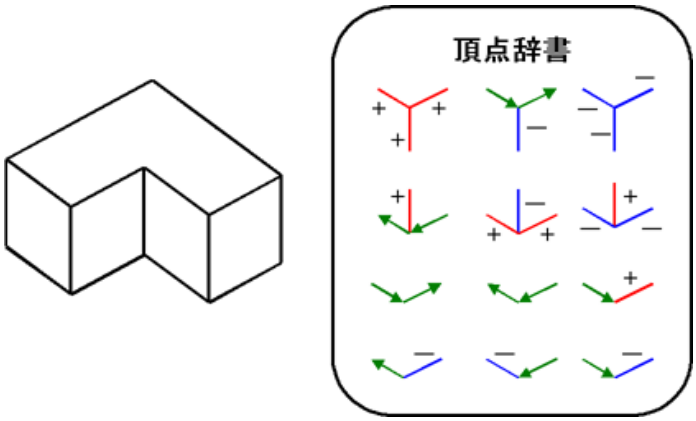


図 2.11: 頂点辞書を使用することで稜線の凹凸判定ができる

2.2.5 エネルギー最小法による 3 次元形状決定

一般に 2 次元の投影図から 3 次元の形状を一意に定めることはできないが、たとえば図 2.12のような投影図から人間が想像する 3 次元形状は、ほぼ一意である。[Pentland89] は、この事実に注目し、 2 次元の曲線を 3 次元に復元する際のエネルギー関数を作成し、その値を最小にするような 3 次元形状を生成することで、比較的人間がイメージする形状と近いものを構築することが可能であることを示した。

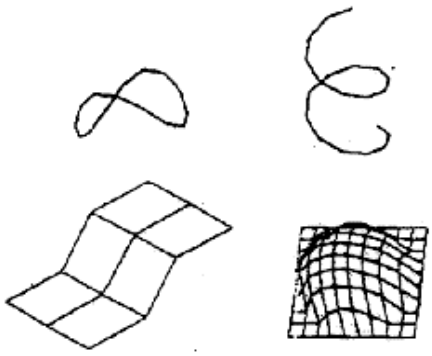


図 2.12: 2 次元のイラストに対して人間がイメージする 3 次元曲線 [Pentland89]

2.2.6 スケッチインターフェースによる入力線分の修正

[松田 99]、 [Baudel94] は、紙面上にスケッチを描くような試行錯誤による線分の入力を可能にした。これは、既に描かれた直線や曲線に対し、線分の重ね書きによって修正を施す手法である (図 2.13)。本研究では、この逐次清書法を折れ線に対して適用する手法を提案している。詳細は 3.4.2節に記す。

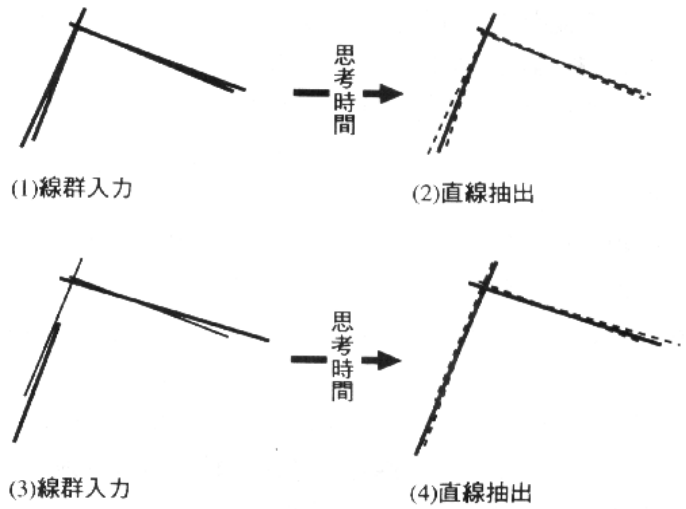


図 2.13: 直線および曲線の逐次清書法

[松田 99]

2.2.7 ファジー理論を用いた入力形状の推定

[Chen96] は、ファジー理論を用いることで、ユーザが入力した線分が、直線、円、円弧、楕円、楕円弧、その他のいずれを表しているかを認識するアルゴリズムを提案し、高い認識率を達成した。提案されている手法では、上記の線分しか認識できないが、 3 次元形状の認識にファジー理論を応用することが可能であれば、スケッチ入力からの形状構築に役立つものだと考えられる。

2.3 計算機を用いたモデル表示手法

計算機内に構築された 3 次元形状を視覚的に確認するために、ディスプレイ上への形状の表示が行われる。計算機を用いて形状を表示する際には、図 2.14のような陰影表示を行うのが一般的であるが、このような単純な陰影表示では表現力に乏しく、無機質な表示となりがちである。そのため、形状の構築が終了したのち、表現力を高めるためのレンダリング処理を別に行うのが一般的である。レンダリング処理では、光学カメラで物体を撮影したのと同じような形状表示を行う**フォトリアリスティックレンダリング**と、それとは異なる**ノンフォトリアリスティックレンダリング**の手法が存在する。

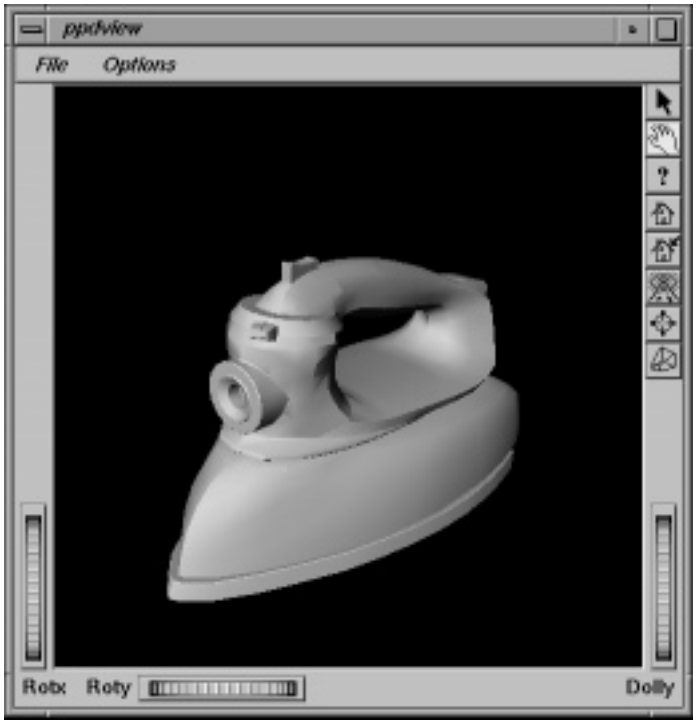


図 2.14: 計算機を用いた形状の陰影表示

2.3.1 フォトリアリスティックレンダリング

長い間に渡り、光学カメラで物体を撮影したのと同じような形状表示を行うことを目標とした、写真画質をめざしたレンダリング手法に関して、多くの研究がなされてきた。近年では厳密な光学モデルの確立と、高速な計算機の出現により、ほとんど写真と区別できないクオリティの画像を正確に、かつ短時間で得ることが可能となっている。 [Keller97] は、ラジオシティを用いた写真画質の出力を高速に行う手法を提案した (図 2.15)。

図 2.15: ラジオシティを用いたレンダリング
[Keller97]

2.3.2 ノンフォトリアリスティックレンダリング

写真と同じような画質の出力が可能となった一方で、必ずしも写真による形状の表現が最良ではないことも指摘され、近年では、アニメ風にレンダリングする手法や、敢えて曖昧な輪郭を用いることで人間が手で描いたように描画を行う研究がなされている。このようなレンダリング手法をノンフォトリアリスティックレンダリング (Non-photorealistic Rendering (NPR)) と呼ぶ。

[Gooch98] は、機械部品の形状を把握しやすくするために、その表示において色の扱いを工夫したり輪郭線を強調して表示する手法を提案している (図 2.16)。

また、 [Rademacher99] は見る角度によって対象物の形状を変化させ、かつアニメ基調での表示を行うことで、デザイナの意図した表現を実現させている (図 2.17)。

アイデアの発想段階においては、発想を妨げないように、従来の紙に行っていたスケッチと同じように表示されることが望まれることから、 [C.Zeleznik96] や [Igarashi99] においては、輪郭線を曖昧に表示するスケッチ風のレンダリングが用いられている。

また、 [Markosian97] は、輪郭線と特徴線のみを高速に描画する手法を考案し、これを用いて、リアルタイムに NPR を適用する手法を提案した。これにより、スケッチのように描画されたオブジェクトを、インタラクティブに操作することができる (図 2.18)。

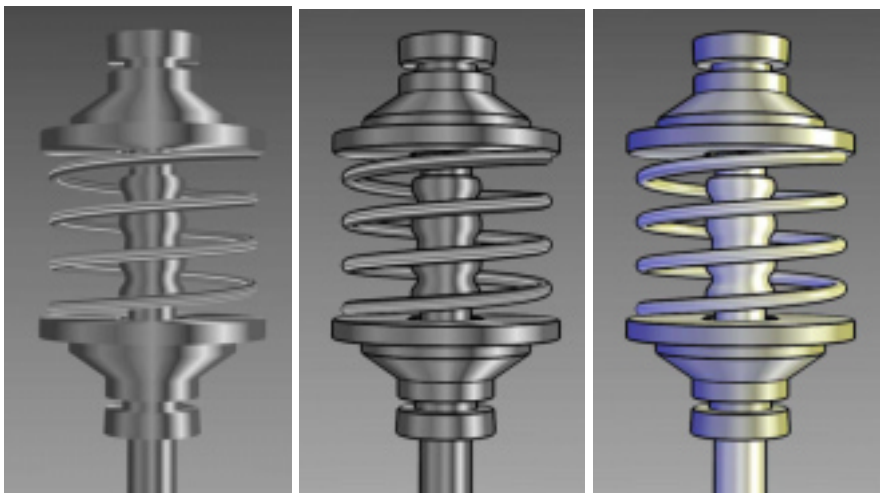


図 2.16: 輪郭を強調した表示

[Gooch98]

図 2.17: アニメ基調の表示

[Rademacher99]

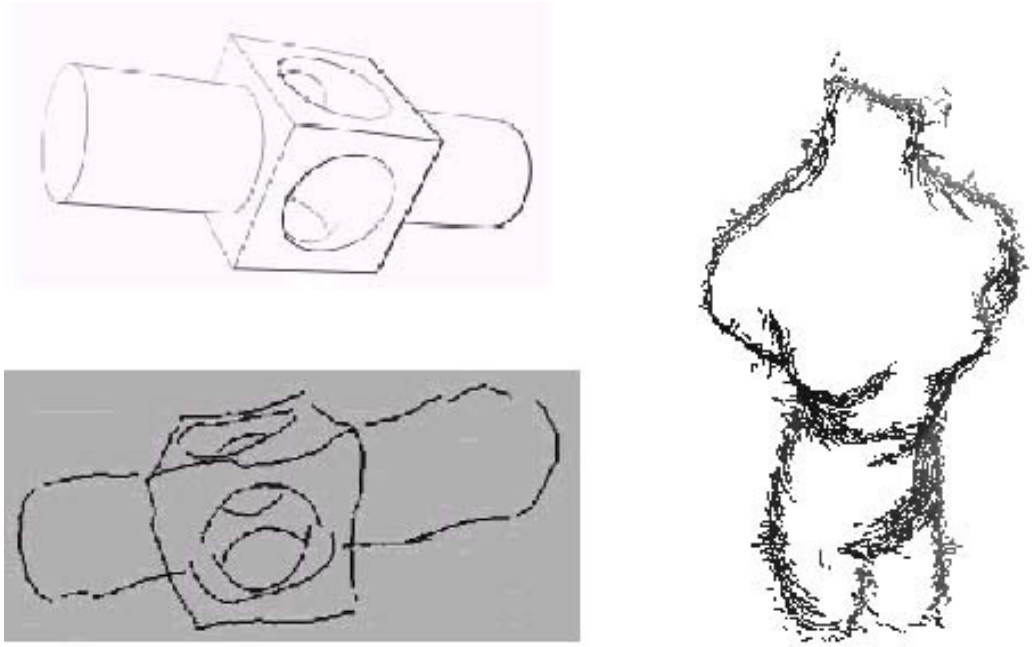


図 2.18: 手描き風のリアルタイムレンダリング
[Markosian97]

第 3 章

3 次元スケッチ

第 3 章

3 次元スケッチ

前章では 3 次元形状を計算機で構築するための手法と、その表示手法についてまとめた。本章では、それら各手法の特徴を考慮した上で、本研究で提案する 3 次元スケッチの具体的な手法について説明する。まず、本手法で対象とするデザインについて述べ、大まかな流れを示した後、手法の詳細について説明する。

3.1　対象とするデザイン

本研究では、図 3.1のような一般的な家電製品にみられる形状を、デザイナの単一スケッチ図から構築する手法を考える。

前章では様々なモデル構築手法についてまとめたが、この目的に近いものには、以下の 3 つの手法があった。

(a) ジェスチャーを用いてプリミティブを生成し、それらをアセンブリする手法 [C.Zeleznik96]

(b) ストロークを直接利用して 3 次元曲線を生成し、それを元に曲面を生成する手法 [古島 93]

(c) ストロークによって描かれた輪郭線から自動的に球体を生成し、それに凹凸部を追加してゆく手法 [Igarashi99]

しかし、 (a) の手法ではプリミティブ形状しか生成できず、デザイナのストロークを活かした形状が作れないという問題がある。

また (b) の手法では、 1 つのスケッチ図から見ることができない裏側については、別の角度から見たスケッチの追加によって面を生成してゆく必要があるため、直接ソリッドモデルを構築することができない。

(c) の手法では、ぬいぐるみのような形状であれば迅速に構築できるが、工業製品のように、角やフィレットなどの特徴線で構成された形状を構築するのが困難である。

そこで本研究では、デザイナのストロークを直接利用し、特徴線で構成されるソリッドモデルを単一のスケッチ図から生成する手法を考える。

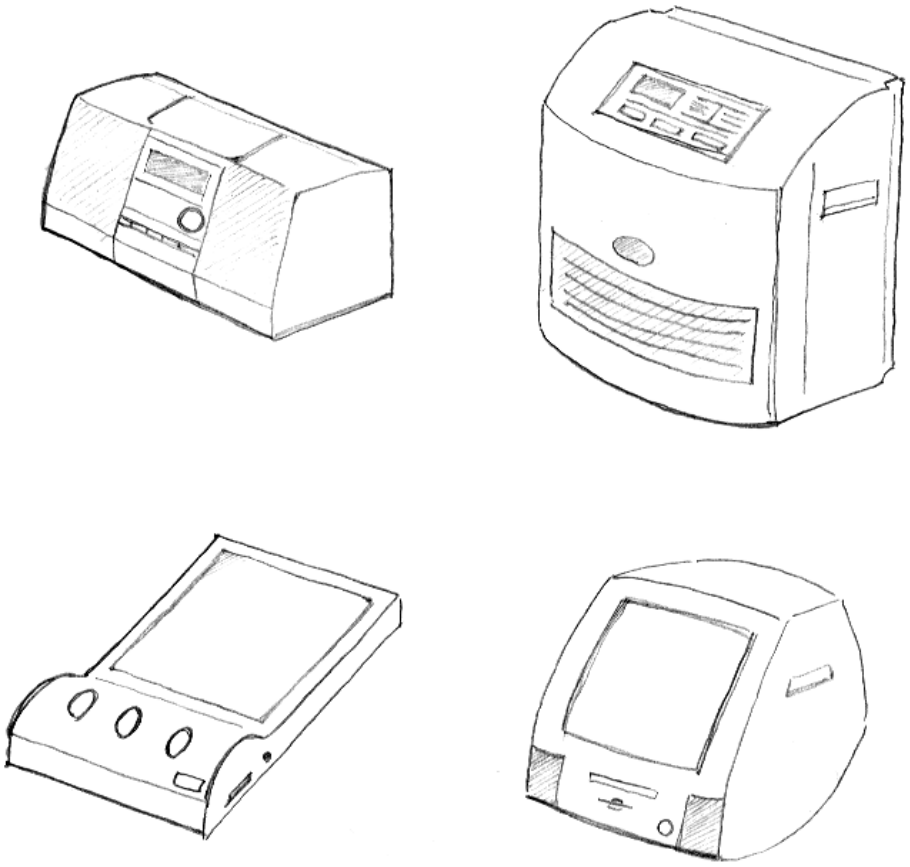


図 3.1: 対象とするデザイン

表示に関してはノンフォトリアリスティックを用いる研究が多くなされているが、デザイナのストロークを活かした表示を行うものは無かった。そこで本手法では、スケッチによって入力されたデザイナのストロークを表示に活かす手法を新たに提案する。

3.2 3 次元スケッチの流れ

本研究で提案する 3 次元スケッチの、入力から出力までの流れを図 3.2に示す。図の上段はユーザのインターフェース部分であり、下段はユーザからは見えないシステムの処理部分である。システムはスケッチの解析、および 3 次元モデルの構築と表示までの処理を、ユーザによる介入無しで行う。そのため、ユーザの視点からは、従来のデザイン作業と変わらないインターフェースによって、描画した形状をそのまま 3 次元形状として扱えるようになる。また、3 次元形状の表示には新しく提案する手描き

レンダリングを適用することで、別の角度から見ても入力されたスケッチスタイルで形状の表示が行われる。具体的な処理の流れは以下のようになる。

(a) ユーザによるスケッチの入力

(b) スケッチからの線画取得

(c) 線画の位相解析

(d) 3 次元形状構築に必要な情報の抽出

(e) 3 次元情報の取得

(f) 3 次元ソリッドモデルの構築

(g) 生成されたモデルへの手描き表現の適用

(h) 手描き表現されたモデルに対する操作

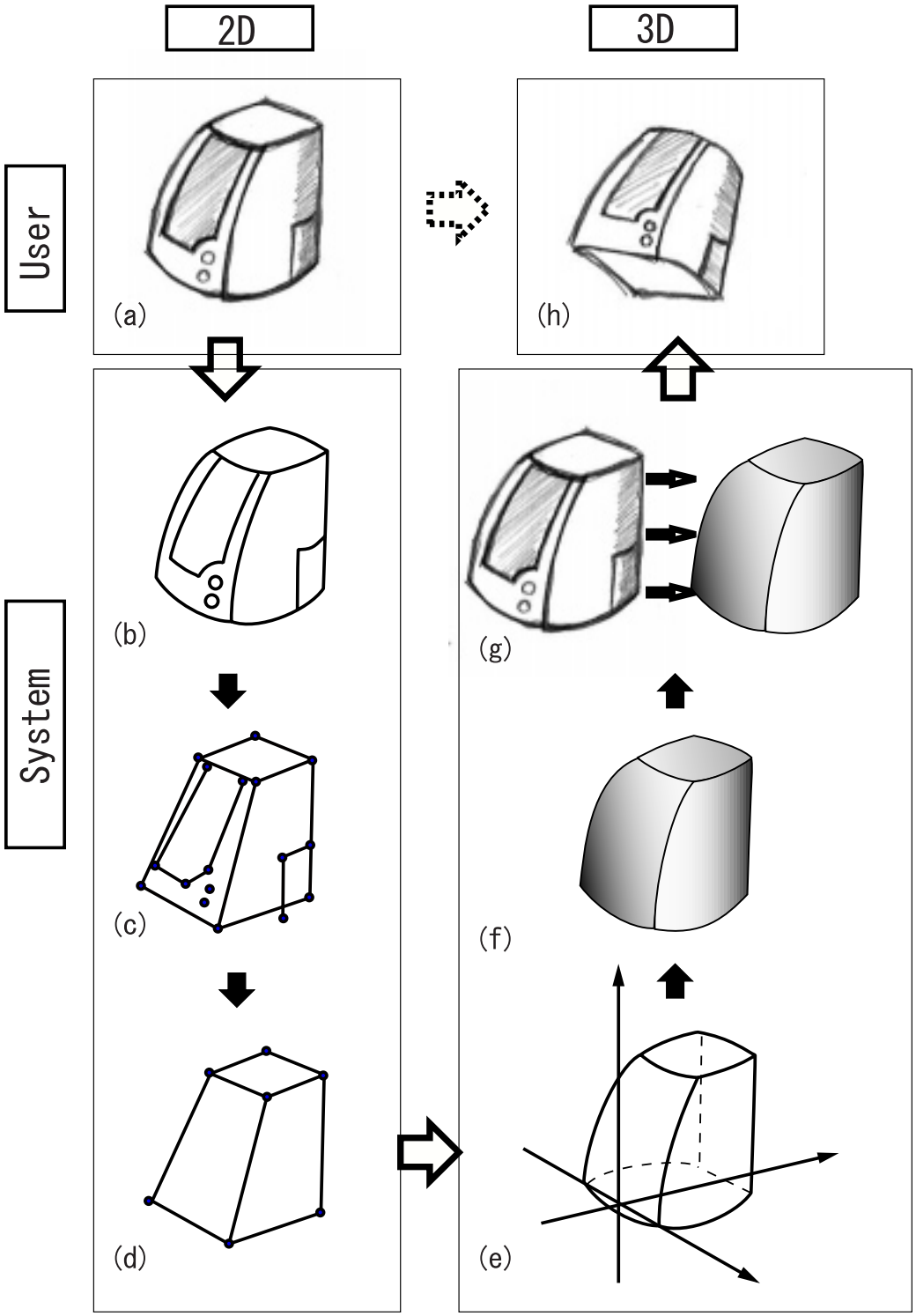


図 3.2: スケッチ入力から 3 次元モデル表示までの流れ (スピーカーのデザインの例)

(a)：ユーザによるスケッチの入力
(b)：スケッチからの線画取得
(c)：線画の位相解析
(d)： 3 次元形状構築に必要な情報の抽出
(e)： 3 次元情報の取得
(f)： 3 次元ソリッドモデルの構築
(g)：生成されたモデルへの手描き表現の適用
(h)：手描き表現されたモデルに対する操作

3.3 入力可能な形状の制限

一般にデザイナによって描かれるスケッチ図は、対象とする 3 次元形状を 2 次元の紙面へ投影したものであるため、スケッチから 3 次元形状を再構築するには、明らかに情報が不足する。人間は不足する情報をそれまでの経験によって補うことが可能であるが、計算機によってスケッチ図から 3 次元形状を再構築するには、なんらかの前提条件や、さらなる情報の追加が必要となる。

本研究では、入力可能な形状に対して前提条件を設け、この前提条件によって得られる情報をスケッチ図に付加することで、単一のスケッチ図から 3 次元形状を構築できるようにする。具体的には、図 3.2の流れ図で示したような左右対称な六面体形状を入力可能な形状として制限する。

ここで述べる、入力可能である左右対称な六面体形状とは、以下の条件を満たすものとする。

- 4 辺に囲まれた曲面を 6 つ持ち、左右対称な形状である。
- 背面、および床に接する底面は平面である。

描画に関する条件は次の通りである。

- 斜め上方向から描かれたもので、 3 つの面が見えている。

本手法では、上記のような条件を満たす形状について 3 次元化を行う。具体的な 3 次元化手法については次節以降で述べる。形状が制限されてしまうが、一般的な家電製品などにおいては、左右対称で、かつ 6 つの面に囲まれたものが多数存在する為、図 3.1に示すようなものを入力可能である。

形状の制限と引き替えに得ることができるメリットには、次のようなものがある。

- 異なる視点からの投影図を描画する必要がなく、単一のスケッチから 3 次元形状を構築できる
- 位相が既知であるため、スケッチから形状の推測を行う必要がなく、アルゴリズムを単純化できる
- スケッチには描かれない、裏側の見えない部分に関しても形状を生成できるため、直接ソリッドモデルを構築できる
- 視点位置を任意の位置にとることができ、好みのパースでスケッチを行うことができる

また、今回は 3 次元スケッチの有効性を示すことが目的であるため、入力可能な形状が限られているが、本節以降で紹介する 3 次元化の手法と同じ枠組で、入力可能な形状のバリエーションを増やすことが可能である。

3.4 スケッチ入力からの線画抽出

スケッチから 3 次元形状を構築するためには、入力されたスケッチを解析し、形状に関する情報を抽出する必要がある。本節では、スケッチによる入力から、形状の特徴を表す線画を抽出し、解析を行うまでの手法を述べる。

3.4.1 ユーザによるスケッチの入力

スケッチを入力とする場合、従来のスケッチスタイルを維持する最良の方法は、紙に描かれたスケッチ図をスキャナなどで計算機に取り込みそれを解析することであろう。しかし、このようにして得られる情報は、画像情報に過ぎないため、効率良く形状の特徴を表す線群を抽出することは非常に困難である。

一方、タブレットやマウスなどによって計算機上に直接スケッチを描くのであれば、デバイスの動きから時系列に並んだ点群を得られるため、そのまま線群の情報を得る事ができる。この線群から形状特徴を表す線を選択すればよいため、必要な処理はスケッチをスキャナなどで取り込む方法に比べると、格段に少なくなり効率的である。しかし、この方法は、デザイナが普段使い慣れないマウスを使うという点で、インターフェースに問題がある。

そこで、本研究ではディスプレイにデジタルペンで直接スケッチを描く事ができる液晶タブレット (WACOM 製 PL-300[WACOM]: 図 3.3) と、ノートに描かれたストローク情報を保存する事ができる CrossPad [IBM] (図 3.4) をスケッチの入力に用いた。いずれもマウス入力よりも柔軟性があり、紙とペンを用いたインターフェースと同じような入力が可能である。

CrossPad は任意の紙の上にボールペンを用いてスケッチを行う事ができるため、デザイナに負荷を与えずにストローク情報を得る事ができる。しかし、スケッチ図を計算機に取り組むのに若干の手間が必要であるため、スケッチの描画が終了次第、すぐに 3 次元形状として確認するのには、液晶タブレットの方が適している。

液晶タブレットや CrossPad で得られる情報は、基本的にマウス操作で得られる情報と同じである。ペンが画面 (紙面) に対して接している状態で移動されると、それがドラッグ操作として、一定間隔にその点座標値を得る事ができる。

このようにして、一回のドラッグ操作で得られる点列からなる折れ線を**ストローク** (図 3.5(a)) と呼ぶこととする。

ところで、スケッチにおいて輪郭線などの特徴線は、何本ものストロークを重ねて表現されることが多い [松田 99]。そこで、 1 本の特徴線を表現する、 1 本または複数本のストロークの集合を**ストローク集合線** (図 3.5(b)) と呼び、ストローク集合線によって表される特徴線を**芯線** (図 3.5(b) 中の太い破線) と呼芯線は明示的に入力されるものではない。

図 3.3: 液晶タブレット

[WACOM]

図 3.4: CrossPad

[IBM]

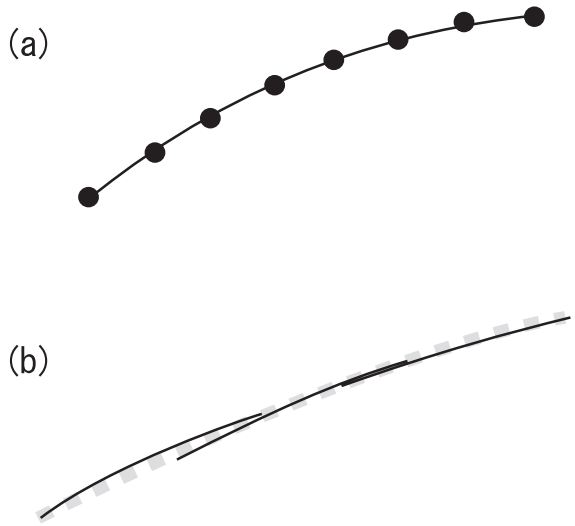


図 3.5: ストローク (実線) と芯線 (破線)

3.4.2 芯線の取得

芯線は複数本のストロークを代表し、形状の特徴を表す線である。そのため、新しいストロークが入力されることによって、芯線を更新する必要が生じる場合がある。例えば、図 3.6(a) に新しいストロークが (b) のように追加された場合、 (c) のように芯線の更新を行う必要がある。

ストロークが入力されるたびに既に存在する芯線との位置関係を逐次判定し、この更新処理を必要に応じて行う。

液晶タブレットでストロークが入力されるときには、リアルタイムで行い、 Cross-Pad で入力されるときにはストローク情報を計算機に取り込み、それを入力順に再生することによって処理を行う。

[松田 99] は、曲線について逐次更新を行う手法を提案している、本研究では線分を折れ線として扱う為、そのための新しい手法を提案する。

まず、新しいストロークが入力された時に、すでに存在する芯線との位置関係を図 3.7(a) のように表す。ここで、太線が既存の芯線、細線が新しく入力されたストロークである。

この時、既存の芯線と新しいストロークの位置関係を図 3.7(b) のようにパラメータ d_1, d_2, α, l を用いて表す。芯線の端点を P、ストロークの端点を Q と、それぞれの端点から他方へ降ろした垂線の足を P', Q' とすると、 d_1, d_2 はそれぞれの $|PP'|, |QQ'|$、 α は PQ' と $P'Q$ のなす角、 l は点 P' と Q' 間の距離である。

ここで、 d_1, d_2, α の値が小さく、 l の値が大きいほど、新しく描かれたストロークは同じ芯線を構成するものである可能性が高く、その逆であれば、新しいストロークは別の芯線を構成するものである可能性が高い。そこで、それぞれの値に閾値を設け、

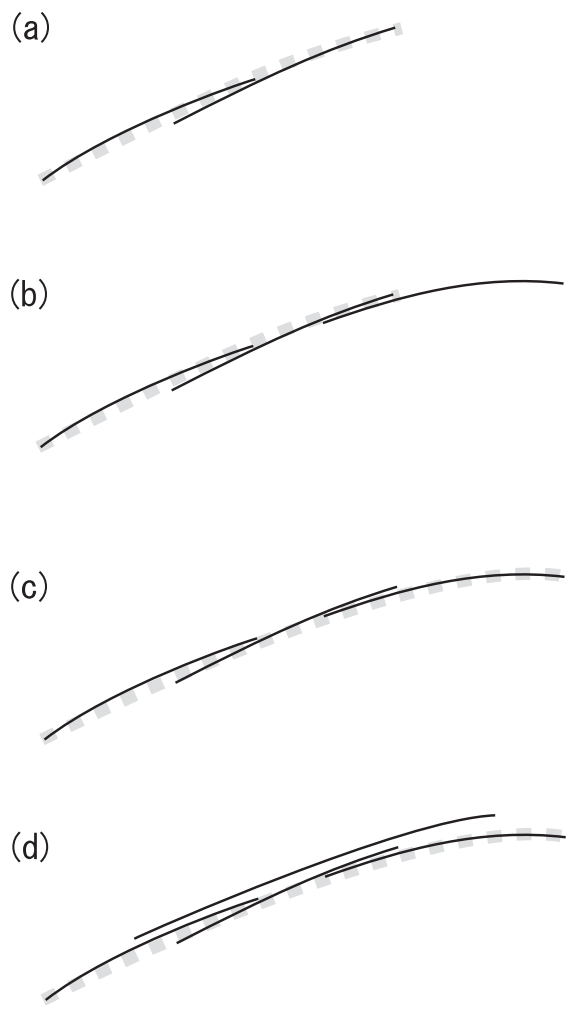


図 3.6: ストローク (実線) の追加による芯線 (破線) の更新

新しいストロークが芯線を更新するものか、そうでないかを判定する。既存の芯線を更新するものであると判定された場合は図 3.7(c) のように既存の芯線とストロークの接続を行なうことで、芯線の更新を行う。図 3.6(d) のように、ストロークの両端点が既存の芯線の 1 つに近い場合は、芯線の更新は行わない。そうでない場合、入力されたストロークそのものを新しい芯線として扱う。

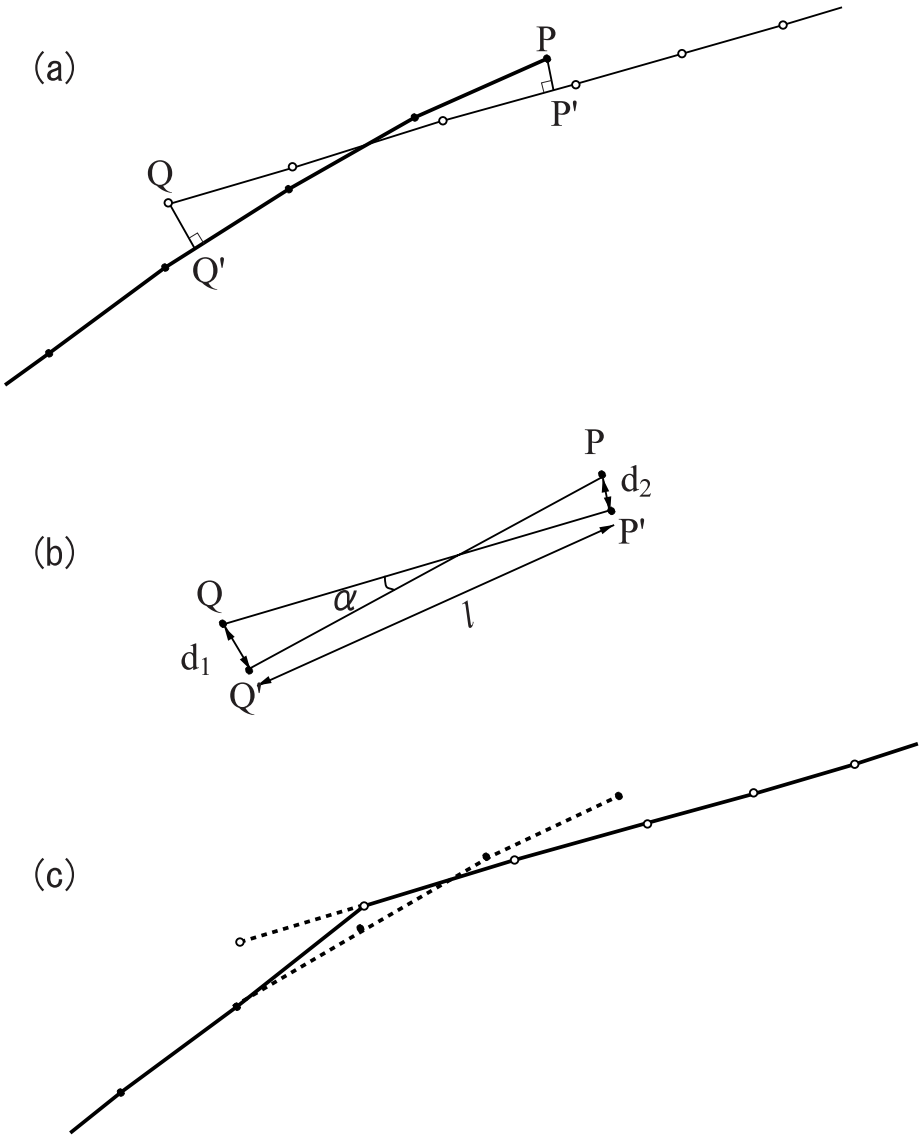


図 3.7: 芯線の更新

3.4.3 芯線のグラフ表現

スケッチ図から抽出した芯線から、さらに位相情報を抽出する為に、芯線をグラフで表現する。

グラフの構成要素である辺は芯線に、頂点は芯線の両端点が対応する。グラフの構

築は、スケッチの入力が終了した段階で行う。

入力された芯線の端点は一般に他のどの芯線の端点とも完全に一致することはないため、閾値以下の距離内に存在する端点同士を結合し、同一頂点とみなす。

データは、辺から両端点に対応する頂点への参照と、頂点から自身を含む辺への参照リストとして保持する。こうすることで、芯線どうしのつながり (位相) をグラフで表す事ができる。

3.4.4 グラフの位相解析

ユーザによって入力された複雑なグラフから、実際に 3 次元モデルを構築する際に必要となる辺と頂点のみの抽出を行う。

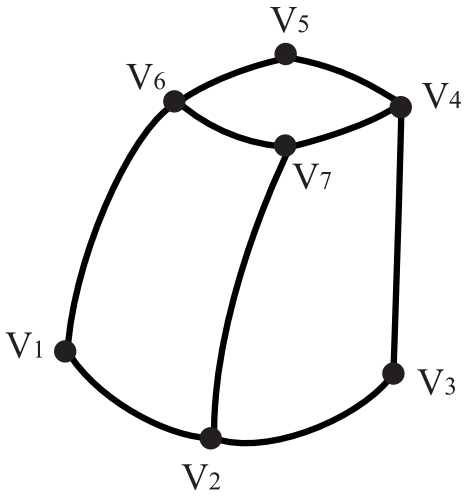


図 3.8: グラフからの形状情報抽出

今回は図 3.8のような左右対称な六面体形状に限定したため、 3 次元形状の構築に必要なのは、 9 本の辺と 7 つの頂点のみとなる。本研究では、以下のようなアルゴリズムでそれぞれの頂点を決定した。

1. 描画領域左上隅に最も近い頂点を V_6 とする
2. 描画領域右下隅に最も近い頂点を V_3 とする
3. 描画領域左辺に最も近い頂点を V_1 とする
4. V_3 に接続する辺を持ち、描画領域上辺に最も近い頂点を V_4 とする
5. V_3 に接続する辺と V_1 に接続する辺を持つ頂点を V_2 とする
6. V_4 に接続する辺と V_6 に接続する辺を持ち、描画領域上辺に最も近い頂点を V_5 とする

7. V_4 に接続する辺と V_6 に接続する辺を持ち、描画領域下辺に最も近い頂点を V_7 とする

3.5 3 次元情報の復元

投影図から 3 次元形状を再構築するには、形状がどのように投影されたかを知る必要がある。一般の CAD システムでは 3 面図などを用いるため、投影パラメータは既知であるが、デザイナが描いたスケッチでは、3 次元空間上でのデザイナの視点位置を推定する必要がある。なお、スケッチ図をデザイナの視点に置かれたカメラによって撮影されたものと想定し、今後デザイナの視点位置を**カメラ位置**ということとする。本節では、はじめに入力形状の拘束条件を使用して取得した芯線からカメラ位置を推定する方法について述べる。その後、カメラ位置情報をもとに 3 次元形状を再構築する方法について説明する。

3.5.1 カメラ位置の推定

3 次元空間内で平行な 2 直線を平面に投影したときに、投影面でその 2 直線が交わる場合、その点 (図 3.9(a) 中 L,R,V) を消失点と呼び、スケッチ図からこの 3 つの消失点を求めることができれば、 [近藤 88] の手法により、投影中心である視点を求めることができる。

消失点は、 3 次元空間内で平行な 2 つの線分を投影したものを延長することで、その交点として求められるため、本手法で入力可能な形状として制限した左右対称な六面体については、互いに対称な位置関係にある点を結ぶ線分から、 1 つの消失点を求めることができる。しかし、求めることができる消失点を x 軸方向とすると、 y 軸、 z 軸方向で平行な直線はスケッチ内に存在するとは限らない為、スケッチ入力から他の 2 つの消失点を直接求めることはできない (図 3.9(b))。

そこで、本手法ではカメラ位置、および消失点位置にいくつかの仮定をおくこととする。まず、次のような方法でスケッチ図から x, y, z 軸、および x 軸方向の消失点を定める (図 3.10)。

1. 2 点 v_1, v_2 を通る直線を x 軸に定める。
2. 2 点 v_1, v_2 の中点を原点とする。
3. 原点を通り垂直な方向を z 軸とする。
4. v_4, v_5 の中点から下ろした z 軸と平行な直線 l_2 と、 v_3 と x 軸の消失点を結ぶ直線 l_3 との交点 h と原点 O を結ぶ直線を y 軸とする。
5. 2 点 v_4, v_5 を通る直線 l_1 と x 軸との交点を x 軸方向の消失点とする。

次に、 x 軸方向の消失点から延ばした水平線と y 軸を延ばした直線との交点が y 軸方向の消失点であるとし、 z 軸方向の消失点は原点から垂直下方向に、カメラ位置は

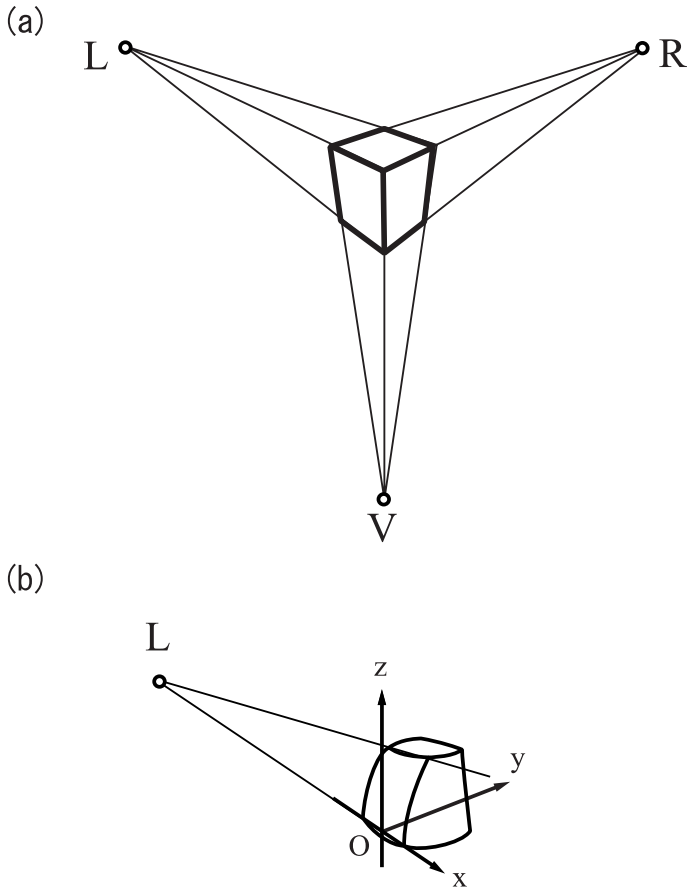


図 3.9: 消失点

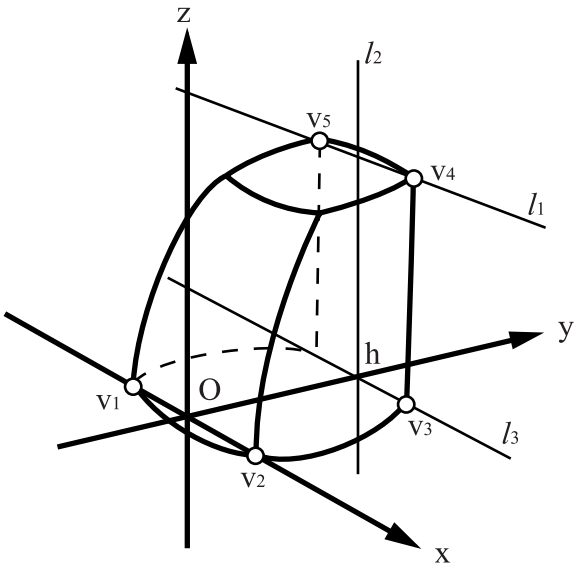


図 3.10: 座標軸の決定

原点 O を通り、紙面に対して垂直方向にあると仮定する (図 3.11)。この仮定は、一般的なスケッチ図の性質と大きく異なるものではなく、このことにより、入力されたスケッチ図からカメラ位置を以下に述べる手法で決定することができる。

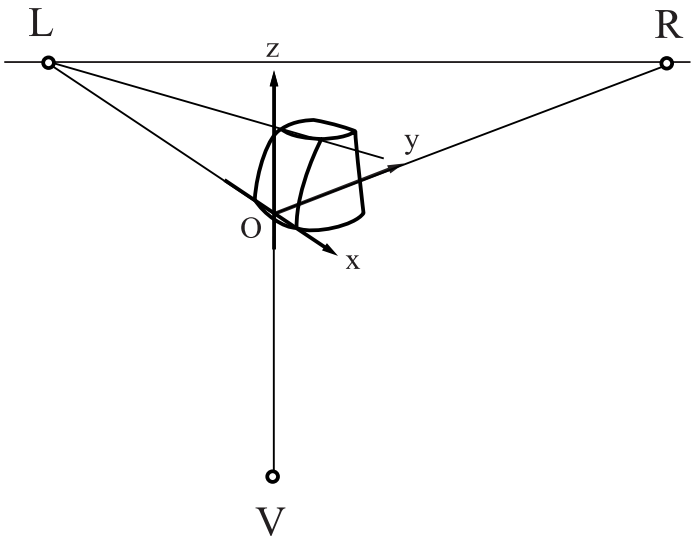


図 3.11: 消失点

図 3.12をもとにカメラ位置を以下のように求める。図中 L は左手方向の消失点 (今回は x 軸方向の消失点)、 R は右手方向の消失点 (今回は y 軸方向の消失点)、 V は垂直方向の消失点 (今回は z 軸方向の消失点)、 O は原点、 H は直線 LR と直線 VO の交点、および E は視点位置である。なお、図 3.12(b) は、スケッチ図の描かれた紙面とデザイナの視点との位置関係を示し、図 3.12(a) は、 (b) を L, R, E を通る平面に垂直上方向から眺めた図である。

この中で、既知のものは L, R, H, O であり、未知のものは E, V である。なお、仮定より E は O に対して紙面垂直方向に存在するため、線分 OE の長さがわかれば視点の位置が定まる。世界座標系の x, y, z 軸は、 3 次元空間において、それぞれ EL, ER, EV に平行であるから、角 LER、角 VEL、角 REV は直角である。また、 EH は LR に垂直であるから、

$$\begin{aligned} EL^2 + ER^2 &= LR^2 \\ EH^2 + HR^2 &= ER^2 \\ EH^2 + HL^2 &= EL^2 \end{aligned} \tag{3.1}$$

よって

$$2EH^2 = LR^2 - (HR^2 + HL^2) \tag{3.2}$$

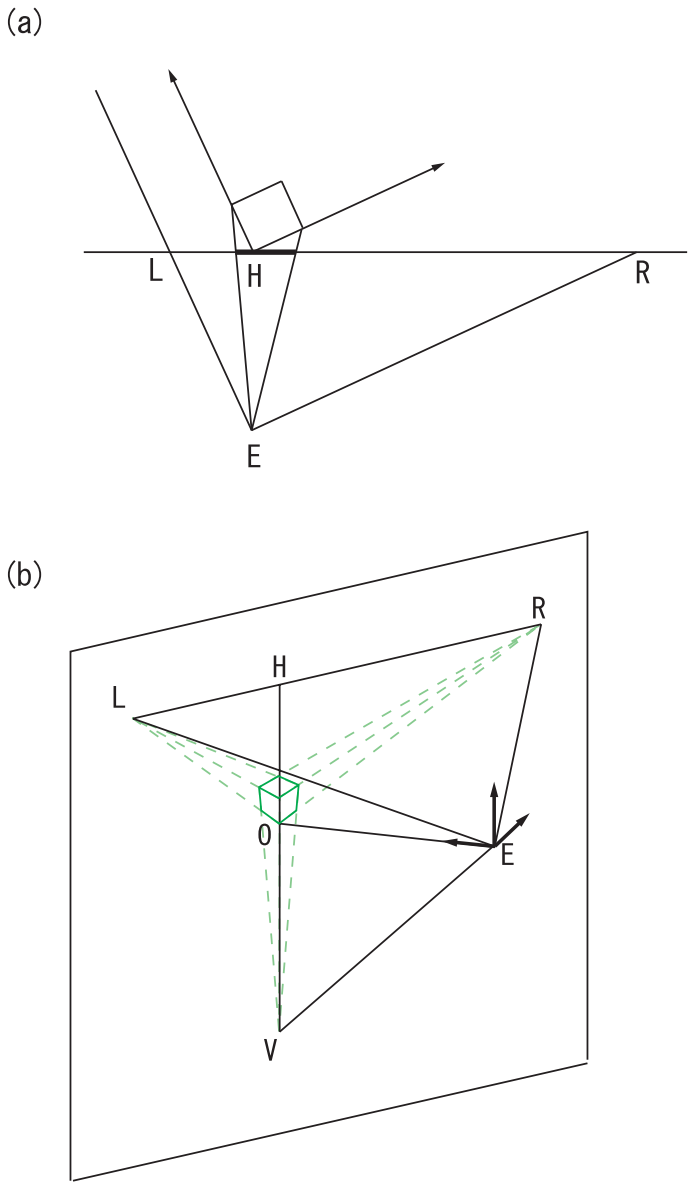


図 3.12: カメラ位置の推定

また、

$$OE^2 = EH^2 - OH^2 \tag{3.3}$$

よって、

$$OE = \sqrt{\frac{LR^2 - (HR^2 + HL^2)}{2} - OH^2} \tag{3.4}$$

以上より、デザイナが入力したスケッチよりカメラの位置を決定することができる。

3.5.2 世界座標からカメラ座標への変換

スケッチ図上の 2 次元座標から 3 次元の世界座標値を得るには、世界座標系と 2 次元の座標系 (画像座標系) の関係を得る必要がある。ここでは、まずカメラ位置が既知であるときに、世界座標系の物体をカメラ座標系へ変換し、画像座標を求める方法[千葉則茂 97] [Trucco98] について述べる。

点 P について、世界座標系での座標値を $\boldsymbol{P}^w = [X^w, Y^w, Z^w]^T$、カメラ座標系での座標値を $\boldsymbol{P}^c = [X^c, Y^c, Z^c]^T$ と表すこととする ($\boldsymbol{P}$ がカメラから可視であれば $Z^c > 0$)。

回転マトリクス $\boldsymbol{R}$ によって座標軸の変換、平行移動ベクトル $\boldsymbol{T}$ によって原点の移動が行われるとすると、 $\boldsymbol{P}^w$ と $\boldsymbol{P}^c$ の関係は次式のように表される。

$$\begin{bmatrix} X^c \\ Y^c \\ Z^c \end{bmatrix} = \boldsymbol{R} \begin{bmatrix} X^w \\ Y^w \\ Z^w \end{bmatrix} + \boldsymbol{T} \tag{3.5}$$

ここで、世界座標系でのカメラの位置を $\boldsymbol{C} = [C_x, C_y, C_z]^T$、カメラ座標の各軸を $\boldsymbol{U} = [U_x, U_y, U_z]^T$、 $\boldsymbol{V} = [V_x, V_y, V_z]^T$、 $\boldsymbol{N} = [N_x, N_y, N_z]^T$ とすると、 $\boldsymbol{T}$ は、

$$\boldsymbol{T} = [-C_x, -C_y, -C_z]^T \tag{3.6}$$

$\boldsymbol{R}$ は、

$$\begin{aligned} \boldsymbol{R}\boldsymbol{U}^T &= [1\ 0\ 0]^T \\ \boldsymbol{R}\boldsymbol{V}^T &= [0\ 1\ 0]^T \\ \boldsymbol{R}\boldsymbol{N}^T &= [0\ 0\ 1]^T \end{aligned} \tag{3.7}$$

$$\boldsymbol{U} \cdot \boldsymbol{V} = \boldsymbol{V} \cdot \boldsymbol{N} = \boldsymbol{N} \cdot \boldsymbol{U} = 0 \tag{3.8}$$

$$\|\boldsymbol{U}\| = \|\boldsymbol{V}\| = \|\boldsymbol{N}\| = 1 \tag{3.9}$$

より、次式で与えられる。

$$\boldsymbol{R} = [\, \boldsymbol{U}^T \boldsymbol{V}^T \boldsymbol{N}^T \,] \tag{3.10}$$

ここで、

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{T} \\ \boldsymbol{O} & 1 \end{bmatrix} \tag{3.11}$$

とすると、

$$\begin{bmatrix} X^c \\ Y^c \\ Z^c \\ 1 \end{bmatrix} = \boldsymbol{M} \begin{bmatrix} X^w \\ Y^w \\ Z^w \\ 1 \end{bmatrix} \tag{3.12}$$

のように、世界座標からカメラ座標への変換を、 4 × 4 マトリックス $\boldsymbol{M}$ によって表すことができる。

3.5.3 カメラ座標から画像座標への変換

カメラ座標系の点 $\boldsymbol{P}^c$ は、ピンホールモデルによる透視変換と、画像座標系への変換によって点 $\boldsymbol{p}_{im} = (x_{im}, y_{im})$ にうつされる。

ここで、 $\boldsymbol{P}^c(X^c, Y^c, Z^c)$ と $\boldsymbol{p}_{im}(x_{im}, y_{im})$ の間に次の式が成り立つ。

$$\begin{aligned} x_{im} &= -\frac{f}{s_x}\frac{X^c}{Z^c} + o_x \\ y_{im} &= -\frac{f}{s_y}\frac{Y^c}{Z^c} + o_y \end{aligned} \tag{3.13}$$

ここで、 f はカメラの焦点距離、 s_x, s_y は画像座標とカメラ座標のスケールファクタ、 o_x, o_y は画像座標での原点を表す。なお、今回は入力がディスプレイから得られたものであるため、 $s_x = s_y = 1$ とできる。

以上より、点 $\boldsymbol{P}$ の世界座標と、カメラマトリックス $\boldsymbol{M}$ が既知である場合、次式によって画像座標系での座標値がわかる。

$$\begin{aligned} x_{im} &= -f\frac{m_{11}X^w + m_{12}Y^w + m_{13}Z^w + m_{14}}{m_{31}X^w + m_{32}Y^w + m_{33}Z^w + m_{34}} + o_x \\ y_{im} &= -f\frac{m_{21}X^w + m_{22}Y^w + m_{23}Z^w + m_{24}}{m_{31}X^w + m_{32}Y^w + m_{33}Z^w + m_{34}} + o_y \end{aligned} \tag{3.14}$$

なお、画像座標とスクリーン座標では y 軸方向の符合を反転する必要がある。

3.5.4 異なる投影図からの 3 次元座標値復元

前節の式を元に、異なる投影図上の対応点と投影図に関するカメラマトリックスが既知であれば、3 次元座標値を以下のようにして求めることができる [古島 93]。

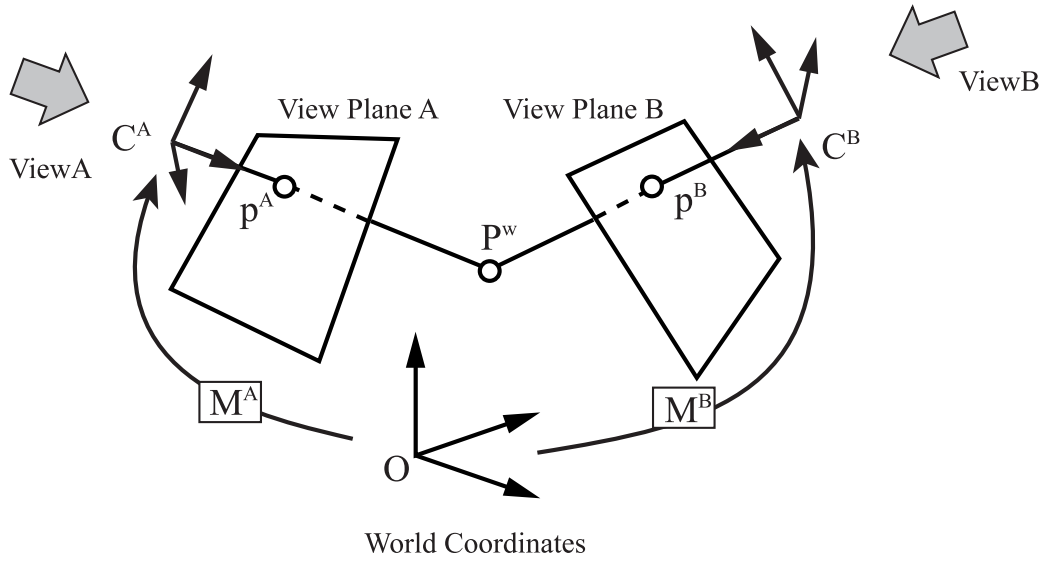


図 3.13: 異なる投影図からの 3 次元座標値復元

カメラ $\boldsymbol{C}^A, \boldsymbol{C}^B$ から、点 $\boldsymbol{P}^w(X^w, Y^w, Z^w)$ を 2 つのカメラマトリックス $\boldsymbol{M}^A, \boldsymbol{M}^B$ を用いて投影した画像座標を $\boldsymbol{p}^A(x^A, y^A), \boldsymbol{p}^B(x^B, y^B)$ とすると、式 3.14より、以下の 4 つの式が成り立つ。

$$
\begin{aligned}
x^A &= -f^A \frac{m^A_{11}X^w + m^A_{12}Y^w + m^A_{13}Z^w + m^A_{14}}{m^A_{31}X^w + m^A_{32}Y^w + m^A_{33}Z^w + m^A_{34}} + o^A_x \\
y^A &= -f^A \frac{m^A_{21}X^w + m^A_{22}Y^w + m^A_{23}Z^w + m^A_{24}}{m^A_{31}X^w + m^A_{32}Y^w + m^A_{33}Z^w + m^A_{34}} + o^A_y \\
x^B &= -f^B \frac{m^B_{11}X^w + m^B_{12}Y^w + m^B_{13}Z^w + m^B_{14}}{m^B_{31}X^w + m^B_{32}Y^w + m^B_{33}Z^w + m^B_{34}} + o^B_x \\
y^B &= -f^B \frac{m^B_{21}X^w + m^B_{22}Y^w + m^B_{23}Z^w + m^B_{24}}{m^B_{31}X^w + m^B_{32}Y^w + m^B_{33}Z^w + m^B_{34}} + o^B_y
\end{aligned}
\tag{3.15}
$$

スケッチ入力から画像座標の点 $\boldsymbol{p}^A, \boldsymbol{p}^B$、およびカメラパラメータが得られるとすると、上式は未知数が X^w, Y^w, Z^w の 3 つ、式が 4 つの連立一次方程式として扱うことができる。

一般に、スケッチから得られる画像座標、カメラパラメータには誤差が含まれる為、4 つの式を厳密に満たす X^w, Y^w, Z^w の解は存在しない。

そこで、

$$
\begin{aligned}
&(m_{31}^A(x^A - o_x^A) + f^A m_{11}^A)X^w + (m_{32}^A(x^A - o_x^A) + f^A m_{12}^A)Y^w \\
&+(m_{33}^A(x^A - o_x^A) + f^A m_{13}^A)Z^w + m_{34}^A(x^A - o_x^A) + f^A m_{14}^A = d_1 \\
\\
&(m_{31}^A(y^A - o_y^A) + f^A m_{21}^A)X^w + (m_{32}^A(y^A - o_y^A) + f^A m_{22}^A)Y^w \\
&+(m_{33}^A(y^A - o_y^A) + f^A m_{23}^A)Z^w + m_{34}^A(y^A - o_y^A) + f^A m_{24}^A = d_2 \\
\\
&(m_{31}^B(x^B - o_x^B) + f^B m_{11}^B)X^w + (m_{32}^B(x^B - o_x^B) + f^B m_{12}^B)Y^w \\
&+(m_{33}^B(x^B - o_x^B) + f^B m_{13}^B)Z^w + m_{34}^B(x^B - o_x^B) + f^B m_{14}^B = d_3 \\
\\
&(m_{31}^B(y^B - o_y^B) + f^B m_{21}^B)X^w + (m_{32}^B(y^B - o_y^B) + f^B m_{22}^B)Y^w \\
&+(m_{33}^B(y^B - o_y^B) + f^B m_{23}^B)Z^w + m_{34}^B(y^B - o_y^B) + f^B m_{24}^B = d_4
\end{aligned}
\tag{3.16}
$$

とし、各 $d_i(i = 1,2,3,4)$ の自乗和

$$
D = (d_1)^2 + (d_2)^2 + (d_3)^2 + (d_4)^2 \tag{3.17}
$$

が最小となるような解を求める。これは、

$$
\frac{\partial D}{\partial X^w} = \frac{\partial D}{\partial Y^w} = \frac{\partial D}{\partial Z^w} = 0 \tag{3.18}
$$

から求まる。

以上より、異なる投影図から 3 次元座標値を復元することができる。

3.5.5 左右対称性を用いることによる単一投影図からの 3 次元座標値の再構築

点 $\boldsymbol{Q}^w$ が、 $\boldsymbol{P}^w$ について YZ 平面に関して面対称であるとすると、

$$
\boldsymbol{Q}^w = \boldsymbol{A}\boldsymbol{P}^w, \ \ \boldsymbol{A} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.19}
$$

が成り立つ。

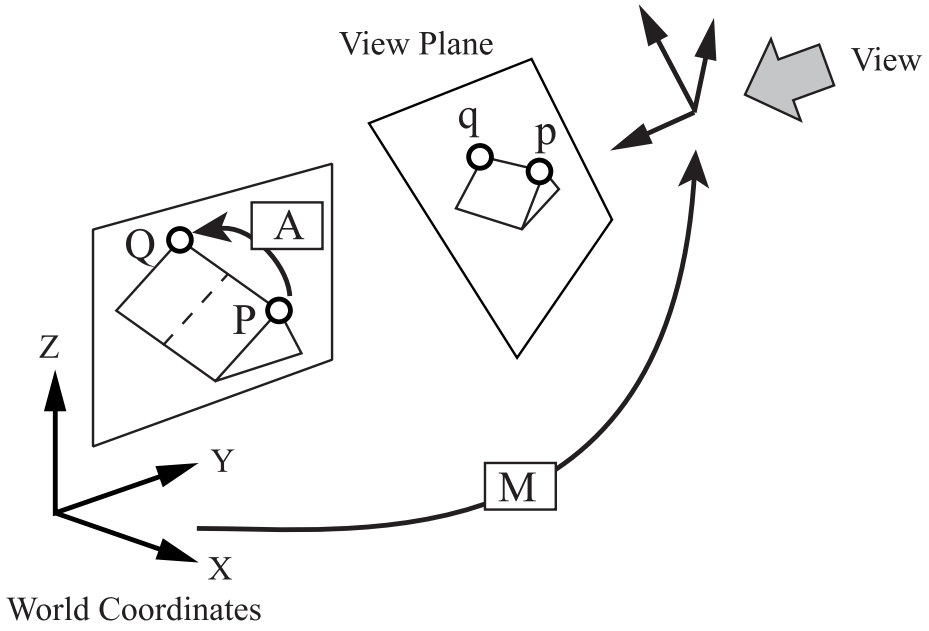


図 3.14: 左右対称性を用いた座標値決定

したがって、カメラマトリックスを $\boldsymbol{M}$ とすると、

$$\begin{aligned} \boldsymbol{Q}^c &= \boldsymbol{M}\boldsymbol{Q}^w \\ &= \boldsymbol{M}\boldsymbol{A}\boldsymbol{P}^w \\ &= \boldsymbol{M}'\boldsymbol{P}^w \end{aligned} \tag{3.20}$$

$$\boldsymbol{P}^c = \boldsymbol{M}\boldsymbol{P}^w$$

と表すことができる。

これは、異なるカメラマトリックスによって同一の点を変換したことと同等の表現であるため、前節で述べたのと同様の手法によって $\boldsymbol{P}^w$ を求める事ができる。点 $\boldsymbol{Q}^w$ は、求めた $\boldsymbol{P}^w$ の x 座標値の符合を反転したものとなる。

一般に、左右対称でなくても、点 $\boldsymbol{Q}$ と点 $\boldsymbol{P}$ の関係を行列 $\boldsymbol{A}$ で表すことができれば、3 次元座標値を求めることができる (点 $\boldsymbol{Q}$ が、点 $\boldsymbol{P}$ の影を表す場合など)。

3.5.6 拘束条件と単一投影図からの 3 次元座標値の再構築

カメラマトリックスが既知であれば、ある点が特定の平面上にあるという拘束条件などから 3 次元座標値を求めることができる。

画像座標 $\boldsymbol{p}_{im}(x_{im}, y_{im})$ から世界座標 $\boldsymbol{P}^w(X^w, Y^w, Z^w)$ を求める場合には、2 つの関係式に未知数が 3 つあるため、たとえば $\boldsymbol{P}^w$ が特定の平面 $aX + bY + cZ + d = 0$ 上に乗っているとすると、次のような連立方程式を解くことで X^w, Y^w, Z^w を一意に求めることができる。

$$aX^w + bY^w + cZ^w + d = 0$$

$$
\begin{aligned}
x_{im} &= -f\frac{m_{11}X^w + m_{12}Y^w + m_{13}Z^w + m_{14}}{m_{31}X^w + m_{32}Y^w + m_{33}Z^w + m_{34}} + o_x \\
y_{im} &= -f\frac{m_{21}X^w + m_{22}Y^w + m_{23}Z^w + m_{24}}{m_{31}X^w + m_{32}Y^w + m_{33}Z^w + m_{34}} + o_y
\end{aligned}
\tag{3.21}
$$

3.6 3 次元ソリッドモデルの構築

前節で述べた手法を用いることで、特定の条件のもとであれば 2 次元の情報から 3 次元座標値を復元することができる。本節では左右対称な六面体形状について 3 次元座標値を復元し、ソリッドモデルを構築する手法について述べる。

3.6.1 六面体を構成する稜線の生成

六面体形状を構成する各稜線、図 3.15中 (1) 〜 (12) について、それぞれの 3 次元座標は、次のように求めることができる。なお、スケッチで入力された各稜線は折れ線であり、生成される 3 次元の稜線も折れ線である。

- (1) と (9), (6) と (8): 互いに対称な折れ線のペア

  それぞれを等しい数の頂点からなる折れ線に近似し、対になる頂点同士の 3 次元座標値を前節で述べた手法で求める (図 3.16)。その後、求めた頂点座標を結ぶ折れ線を得る。

- (2), (5), (7): それ自体が左右対称な折れ線

  中央で分割することで、互いに対称な 2 つの折れ線と同様に扱う。 (2) については、同時に xy 平面上に乗っているという拘束もあるため、一度左右対称な条件で 3 次元座標値を求めた後、 z 軸方向から xy 平面に投影した座標を使用する。

- (3), (4): 平面上への拘束がある折れ線

  折れ線を構成する頂点に対し、特定の平面上に乗っているという条件から 3 次元座標を求める。その後、求めた頂点座標を結ぶ折れ線を得る。

- (10), (11), (12): その他の折れ線

  (11), (12) は、それぞれ yz 平面に関して (4), (3) に対称な折れ線として求める。 (10) は、底面と背面がともに平面であることから、直線となる。

3.6.2 面の生成

ソリッドモデルを構成する各面を、 Coons Patch[Farin97] として生成する。

面の境界となる 4 辺の曲線を $\boldsymbol{c}_1(u), \boldsymbol{c}_2(u), \boldsymbol{d}_1(v), \boldsymbol{d}_2(v), u \in [0,1], v \in [0,1]$ とし、この 4 辺を補間する Coons Patch を $\boldsymbol{x}$ とすると、 $\boldsymbol{x}$ の境界は次の条件を満たす必要がある。

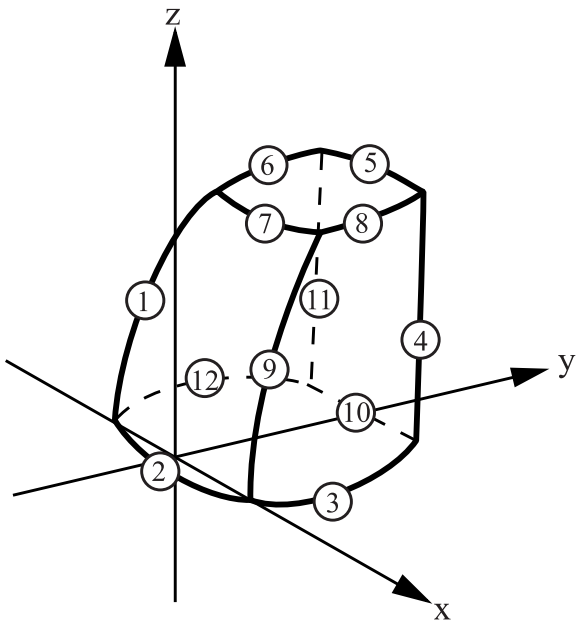


図 3.15: 3 次元モデルの構築

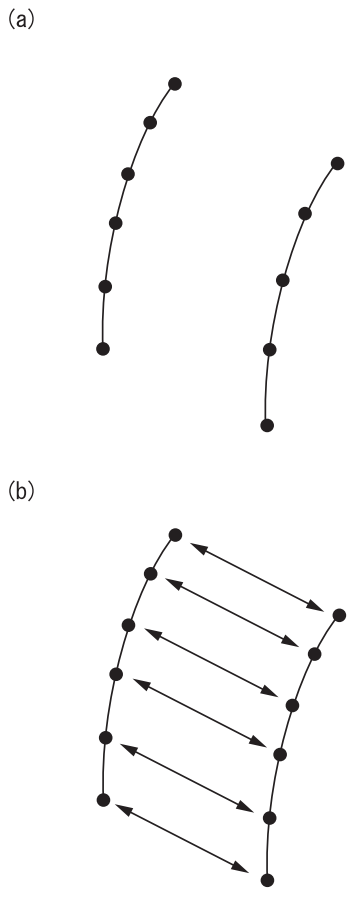


図 3.16: 互いに対称な折れ線を等しい数の頂点を持つ折れ線にする

$$
\begin{aligned}
\boldsymbol{x}(u,0) &= \boldsymbol{c}_1(u) \\
\boldsymbol{x}(u,1) &= \boldsymbol{c}_2(u) \\
\boldsymbol{x}(0,v) &= \boldsymbol{d}_1(v) \\
\boldsymbol{x}(1,v) &= \boldsymbol{d}_2(v)
\end{aligned} \tag{3.22}
$$

ここで

$$
\begin{aligned}
\boldsymbol{r}_c(u,v) &= (1-v)\boldsymbol{x}(u,0) + v\boldsymbol{x}(u,1) \\
\boldsymbol{r}_d(u,v) &= (1-u)\boldsymbol{x}(0,v) + u\boldsymbol{x}(1,v) \\
\boldsymbol{r}_{cd}(u,v) &= \begin{bmatrix} 1-u & u \end{bmatrix} \begin{bmatrix} \boldsymbol{x}(0,0) & \boldsymbol{x}(0,1) \\ \boldsymbol{x}(1,0) & \boldsymbol{x}(1,1) \end{bmatrix} \begin{bmatrix} 1-v \\ v \end{bmatrix}
\end{aligned} \tag{3.23}
$$

とすると、

$$
\boldsymbol{x} = \boldsymbol{r}_c + \boldsymbol{r}_d - \boldsymbol{r}_{cd} \tag{3.24}
$$

によって Coons Patch の面を生成することができる (図 3.17)。

位相が既知であるため、求めた稜線 (1) 〜 (12) から、 6 つの面の境界を求めることができる。本研究では、与えられる 4 つの曲線がそれぞれ 3 次元の折れ線であるため、補間パラメータを折れ線の長さを用いて表し、 Coons Patch は三角形メッシュとして生成した。

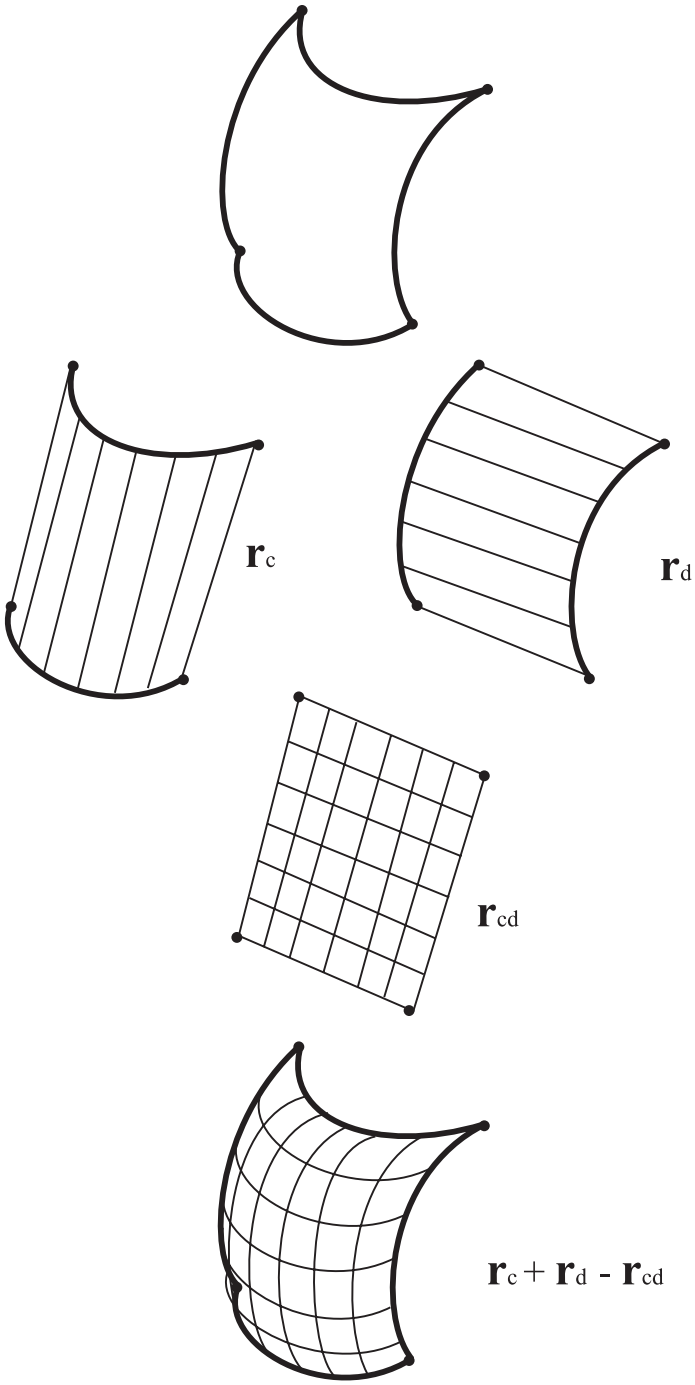


図 3.17: Coons Patch

3.7 ソリッドモデルへの手描き表現の適用

生成された 3 次元モデルをスケッチ入力と同等の表示を行うために、本手法では、スケッチによって入力されたストロークの形状を活かしたレンダリングを行う。前節で述べた手法によって生成されたソリッドモデルに対し、本研究で提案する手描きレンダリングを適用する手法について述べる。

3.7.1 面上へのストロークの配置

本研究では、スケッチによって入力されたストロークを、生成されたモデルの表面上へ配置することで、面上に貼られたテクスチャとして扱う。

生成された 3 次元モデルを、スケッチ図から求めたカメラマトリックスを用いて平面に投影することで、計算機によって生成された 3 次元モデルの投影図とスケッチ図の対応をとることができる (図 3.18)。

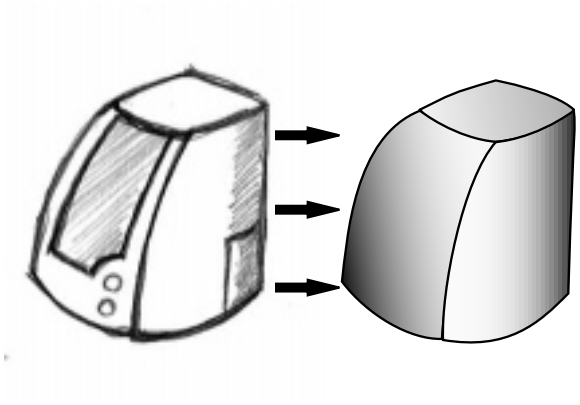


図 3.18: 3 次元モデルの投影図とスケッチの対応

3 次元モデルは三角形メッシュによって構成されているため、各面を投影して得られる三角形に対して、スケッチ入力されたストロークとの干渉判定を行う。ストロークが干渉している面に対し、最も視点位置から手前にある面を求め、ストロークはその面の上に乗っている線分であるとみなす。ストロークは折れ線であるから、それぞれの頂点について、その位置を三角形の重心座標で保存し、面の表示が行われる際に、そのストロークを重心座標を用いて同時に表示する。表示は面単位で行うため、面ごとに折れ線の切断を行っている (図 3.19)。

このようにすることで、物体を回転させたときに、面の上に描かれたストロークとして扱うことができる。

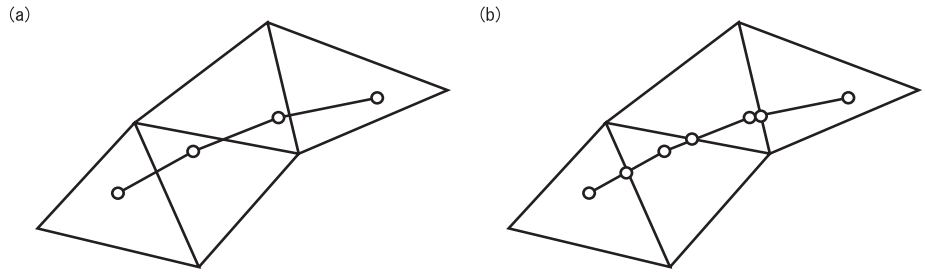


図 3.19: 面の上へのストロークの配置

3.7.2 形状特徴線へのストロークの配置

面の上に描かれた線に関しては、そのまま面の上にテクスチャとして配置することで解決できるが、形状特徴である稜線を表すストロークについても同様に面の上へ配置すると、稜線が輪郭線となるときに、うまく表示されない問題が生じる。例えば図 3.20(a) のように、直方体の稜線を複数のストロークで描画した場合、直方体の面を投影した領域からはみ出している部分に関しては描画が行われないため、 (b) のように稜線が抜け落ちてしまうことが起こる。また、そうでない箇所についても、面上のテクスチャとして扱われているため、ストロークの乗っている面が裏側にまわってしまったときに、 (c) のように描画が行われないという問題が起こる。

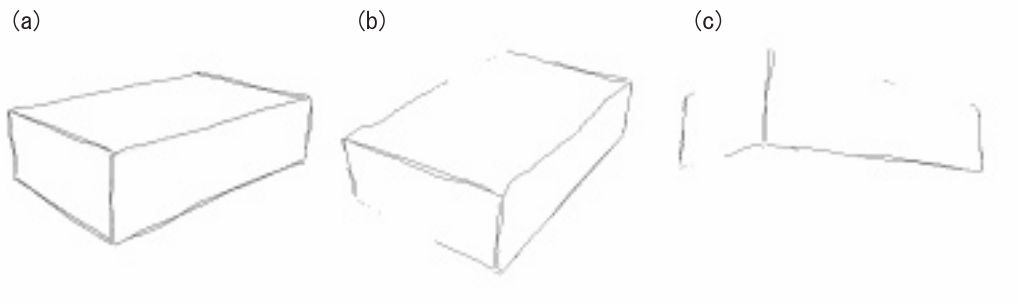


図 3.20: 面の上へストロークを配置した際の問題

そこで本研究では、稜線を表す線は面の上の線とは異なる扱いを行うようにする。具体的には、特徴線として入力されたストロークについて、芯線との相対位置情報を保持しておき、その情報を用いて特徴線の表示を行うようにする。例えば、図 3.21(a) のように描かれた特徴線は、 3 次元形状の回転などによって特徴線の位置が変わった場合でも、図 3.21(b) のように表示されるようになる。つまり、面に対するテクスチャではなく、稜線に対するテクスチャという概念で扱うことになる。

まず、図 3.22(b) を例に、用語を以下のように定義する。

- **Projected Character line(PC)**

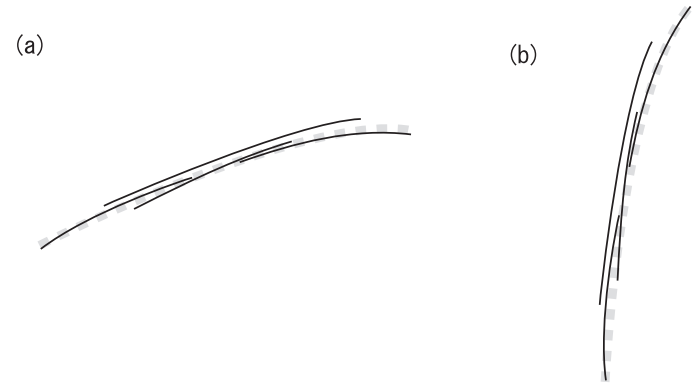


図 3.21: (a) 入力されたストローク (実線) と芯線 (破線)、 (b)3 次元形状の特徴線 (破線) と手描きレンダリングで表示されるストローク (実線)

投影された 3 次元モデルの特徴線

- **Projected Character line Segment(PCS)**

  Projected Character line を構成する各線分要素 (3 次元モデルを構成するメッシュのエッジに相当する)

- **Stroke(S)**

  ユーザによって入力された 1 本のストローク

- **Stroke Segment(SS)**

  Stroke を構成する各線分要素

- **Core line(C)**

  複数のストロークによって表現される 1 本の折れ線 (芯線)

- **Core line Segment(CS)**

  Core line を構成する各線分要素

まず、スケッチ入力時に描画されたストロークの、芯線に対する相対位置情報を取得する。この相対位置情報とは、図 3.22(c) のように、各 SS の両端点ごとに、最も位置の近い CS への参照と、その CS に対する相対位置パラメータ l と t である。l は CS からの距離であり、t は CS の始点を 0、終点を 1 とした場合の、両端点から CS へ下ろした垂線の足の値である。このようにして得られた相対位置情報を、レンダリング時には、投影された特徴稜線のセグメントである PCS に適用することで、芯線に対するストロークの相対位置を稜線に対する相対位置として復元する (図 3.22(d))。

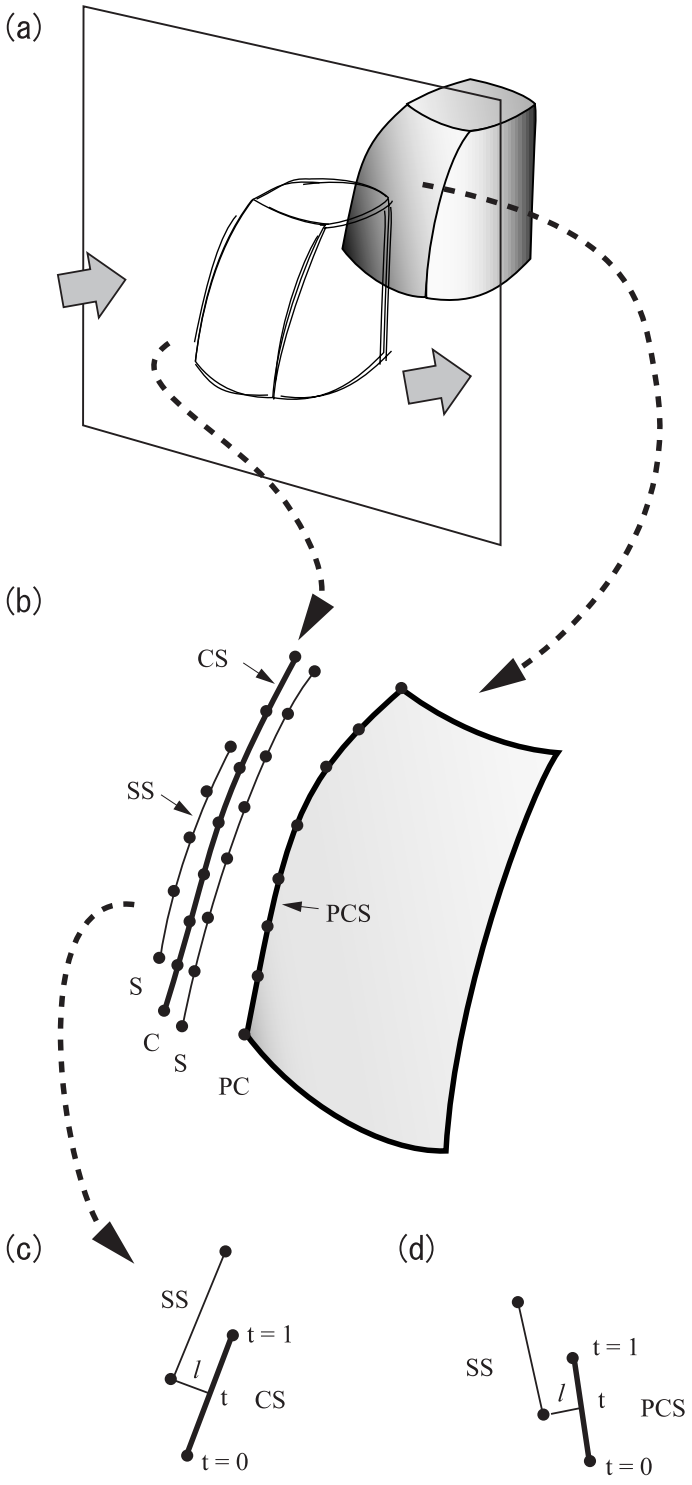


図 3.22: 形状特徴線へのストロークの配置

ここで問題となるのが、どのようにして SS と PCS の対応関係、および相対位置パラメータを取得するかである。A が B の構成要素であることを「A∈B」、A と B が一対一の対応関係を持っていることを「$A \overset{1:1}{\longleftrightarrow} B$」、A と B が多対一の対応関係を持っていることを「$A \overset{n:1}{\longleftrightarrow} B$」と表すこととすると、図 3.22(a) のように、入力スケッチ図と生成された 3 次元モデルの対応をとったときに、以下の相互関係が既知である。

$$\text{PCS} \in \text{PC}$$
$$\text{SS} \in \text{S}$$
$$\text{CS} \in \text{C}$$

$$\text{C} \overset{1:1}{\longleftrightarrow} \text{PC}$$
$$\text{S} \overset{n:1}{\longleftrightarrow} \text{C}$$

求めるものは、

$$\text{SS} \overset{n:1}{\longleftrightarrow} \text{PCS}$$

および稜線の描画時に使用する、各 SS の両端点の PCS に対する相対位置パラメータ l と t である。

まず、特定の PC について、PCS の数を m とすると、PC に対応する C を m 個の CS に分割し直す。すると、次のような対応関係を得ることができる。

$$\text{CS}_i \overset{1:1}{\longleftrightarrow} \text{PCS}_i \ \ (0 \leq i < m)$$

このような分割を行った後、各 SS の両端点について、最も近い CS と、それに対する相対位置パラメータ l と t(図 3.22) を求める。ここで、

$$\text{SS} \overset{n:1}{\longleftrightarrow} \text{CS}_i$$

の関係が求まった。この後、各 SS の相対位置パラメータ l と t はそのままに、参照要素 CS_i を、それに対応する PCS_i に変更する。

以上で、

$$\text{SS} \overset{n:1}{\longleftrightarrow} \text{PCS}$$

と l, t 求めることができた。

レンダリングの際には、PCS が表示されるときに、PCS の代わりに PCS に関連する 1 つまたは複数の SS を相対位置パラメータを使用して描画を行う。

このようにして、特徴線に対しては、2 次元の相対位置でストロークを配置することにより、デザイナのスケッチスタイルを活かしたモデルの表示を行うことができる。

図 3.23は、実際に直方体の稜線に (a) のストロークを配置したものである。 (b)(c) のように、直方体を回転させても、稜線は入力されたストロークを活かした表示が行われているのがわかる ((c) の裏側に関しては、何のストロークも配置されていないため、ここでは表示が行われていない)。

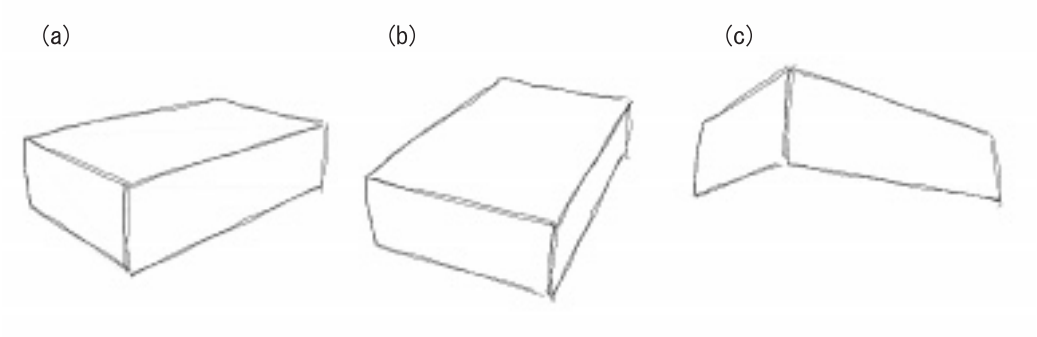


図 3.23: 稜線へのストロークの配置を行った例

3.8 3 次元モデルのデータ構造

本手法で生成した 3 次元モデルは、 3 角形メッシュモデルであるため、生成したモデル (Solid) が、三角形面 (Triangle) と、稜線 (Edge)、および頂点 (Vertex) のベクトルを保持している。三角形面は 3 つの稜線と頂点、およびそれに付随するストロークセグメントを保持している。稜線は、両端点の頂点と両側の三角形面、稜線が形状特徴線であるかを示すフラグ、およびそれ自身が形状特徴線であるときはそれに付随するストロークセグメントを保持している。頂点は、世界座標系と画像座標系の座標値を保持している。 Java 言語でこのデータ構造を表すと次のようになる。

```
class Solid{
  Vector triangles;
  Vector edges;
  Vector vertices;
}

class Triangle{
  Edge edge[3];
  Vertex vertex[3];
  Vector strokeSegments;//segments on this triangle
}
```

```
class Edge{
  Vertex start;
  Vertex end;
  Vector strokeSegments;//segments related to this edge
  boolean cl;//CharacterLine or Not
  Triangle t0, t1;//triangles connected to this edge
}

class Vertex{
  Vector3d worldCoodinates;
  Vector2d imageCoodinates;
}
```

第 4 章

3 次元スケッチを適用した結果と評価

第 4 章

3 次元スケッチを適用した結果と評価

前章で提案した 3 次元スケッチの手法について、計算機上でプログラムを実装した。本章ではこのシステムを用いて、実際にスケッチ入力からの 3 次元形状の構築および手描きレンダリングを行った結果とその評価をまとめる。

実装に当たっては、 Java 言語を用いてプログラムを作成した。使用した計算機は PentiumIII 600MHz, Memory 512MB の PC である。

4.1　3 次元スケッチを適用した結果

実装したシステムを用い、実際にスケッチ図からの 3 次元形状の構築とその表示を行った。まず、形状特徴線からのみ構成される単純なスケッチ図から 3 次元形状の構築を行った結果を示し、次にモデルの構築から手描きレンダリングまでを行った結果を示す。さらに、本システムを用いた応用例を示す。

4.1.1　3 次元形状の入力

本研究で提案した手法には、入力可能な形状が左右対称な六面体形状であるという制限があるが、この条件を満たしていれば 1 枚のスケッチ図から 3 次元モデルの構築が可能となっている。

ここでは、図 4.1と図 4.2に示す 6 種類の形状について、スケッチの入力から 3 次元形状の構築を行った。図中、左の列が CrossPad 上で描画したスケッチ図を計算機に読み込んだもので、中央の列が生成された 3 次元モデルのメッシュ表示である。このようにして生成されたモデルのデータを別の 3D 表示アプリケーションで表示したものが右の列である。

入力したスケッチ図から、意図される 3 次元形状が構築されていることが確認できる。入力から 3 次元モデルの構築および表示にかかる時間はどれも 2 〜 3 秒程度であった。

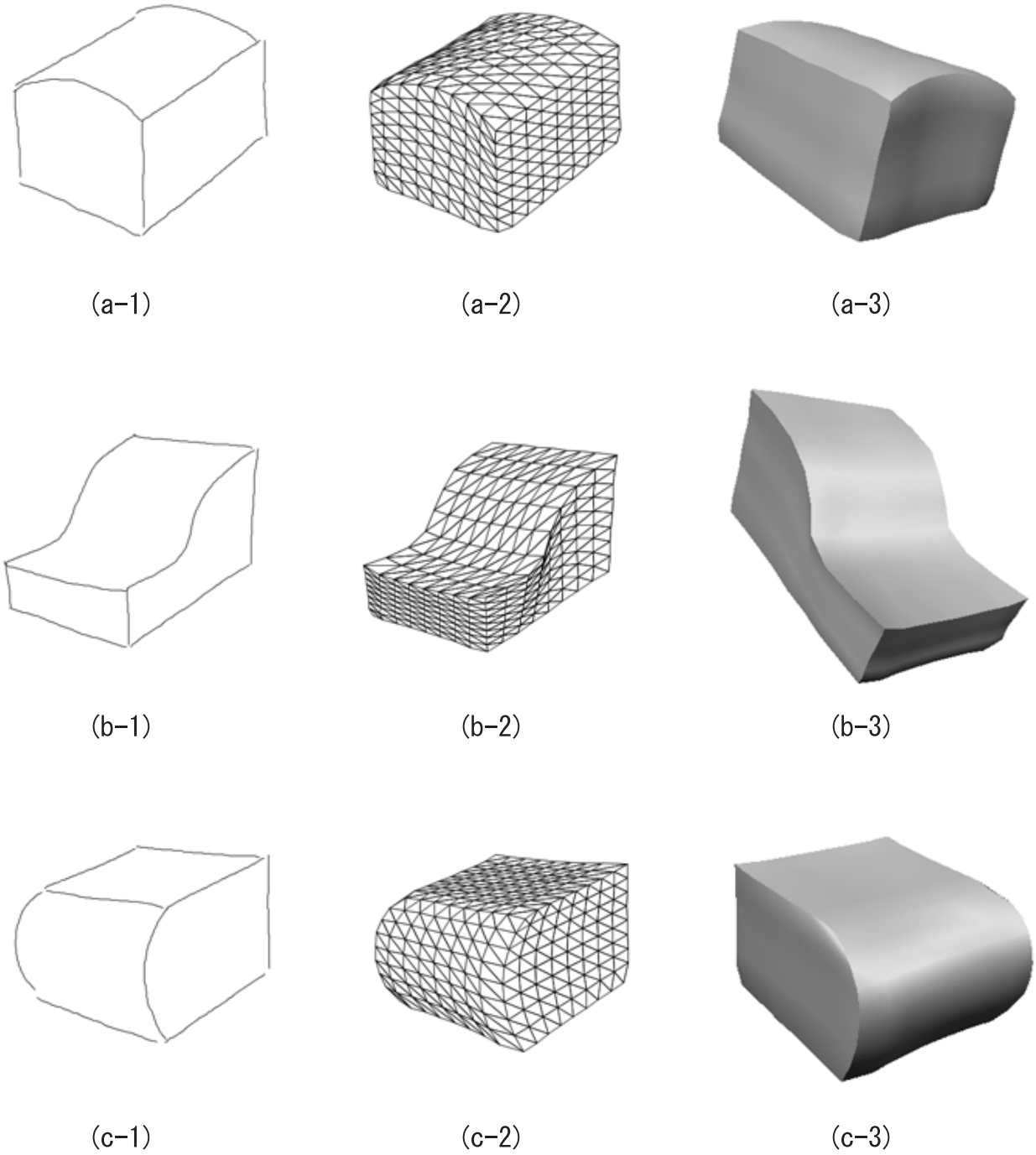


図 4.1: 3 次元形状の入力 (1)

左列: CrossPad 上で描画したスケッチ図を計算機に読み込み表示したもの
中央: 生成された 3 次元モデルのメッシュ表示
右列: 生成されたモデルデータを別の 3D 表示アプリケーションで表示したもの

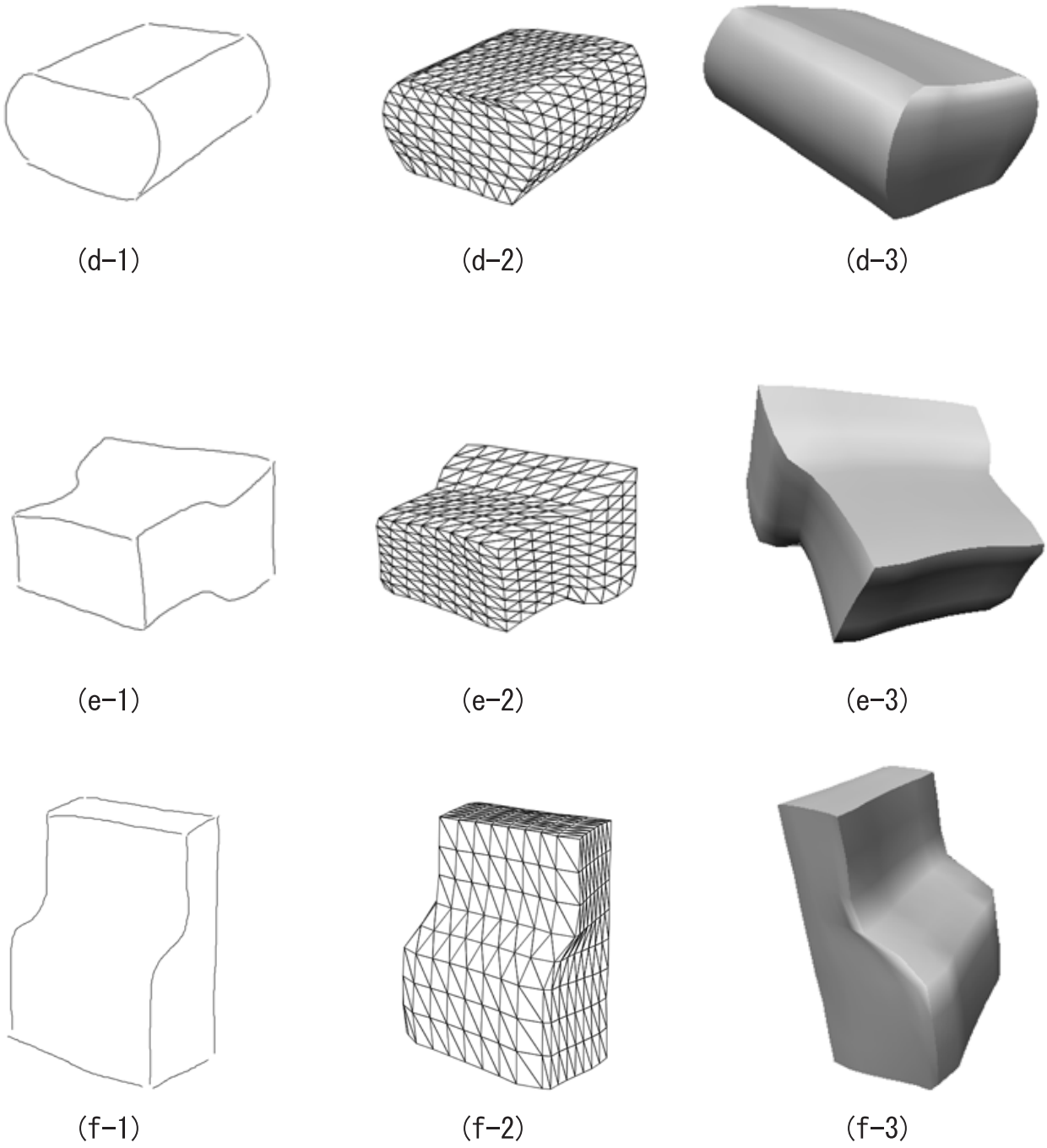


図 4.2: 3 次元形状の入力 (2)

左列: CrossPad 上で描画したスケッチ図を計算機に読み込み表示したもの
中央: 生成された 3 次元モデルのメッシュ表示
右列: 生成されたモデルデータを別の 3D 表示アプリケーションで表示したもの

4.1.2 3 次元スケッチの適用

スピーカーと情報端末のスケッチ図について、本研究で提案する 3 次元スケッチの手法を適用し、モデルの構築から手描きレンダリングまでを行った。

それぞれの適用結果を図 4.3から図 4.6に示す。図中 (a) 〜 (g) が表すものは以下の通りである。

(a) CrossPad 上で描画したスケッチ図を計算機に読み込み表示したもの

(b) 入力されたスケッチ図から抽出された芯線を表示したもの

(c) 図中の 7 つの点は 3 次元形状を決定する稜線を結ぶ頂点。立方体は、芯線をもとに推定されたカメラマトリックスを用いて、単位立方体を紙面上に投影したもの

(d) 構築した 3 次元のソリッドモデルをメッシュ表示したもの

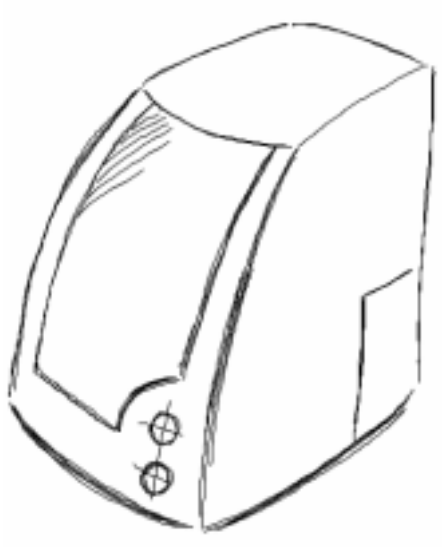
(e) ソリッドモデルに手描きレンダリングを適用して表示したもの

(f) 得られた 3 次元形状を他の 3D 表示アプリケーションで表示したもの

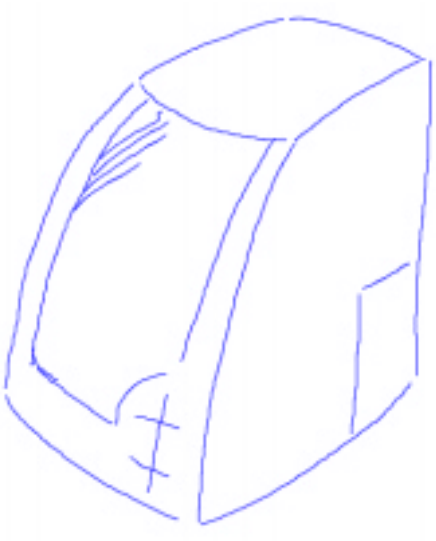
(g) 得られた 3 次元形状を回転させ、別の角度から眺めたもの

それぞれのストローク数、頂点総数、読み込み時間、およびモデル構築時間は下表に示す通りである。ストロークは折れ線として入力されるため、その頂点数の総和を頂点総数として数えている。また、読み込み時間とは、 CrossPad で入力したスケッチ図データを読み込むのにかかった時間であるが、読み込みと同時に芯線の更新を行っているため、その処理に時間がかかっている。読み込みが終わった時点で、芯線の抽出も完了しているため、その後のモデル構築は、ともに数秒の単位で完了している。モデル構築後の、手描きレンダリングは十分高速であり、インタラクティブに回転を行える速度であった。

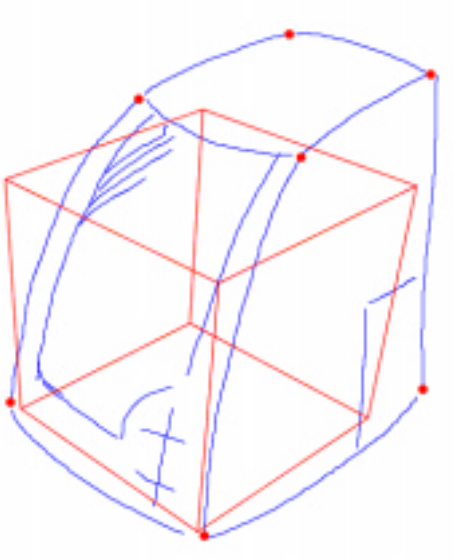
	ストローク数	頂点総数	読み込み時間 (sec)	モデル構築時間 (sec)
スピーカーのスケッチ	68	1941	28	4
情報端末のスケッチ	43	1304	15	4



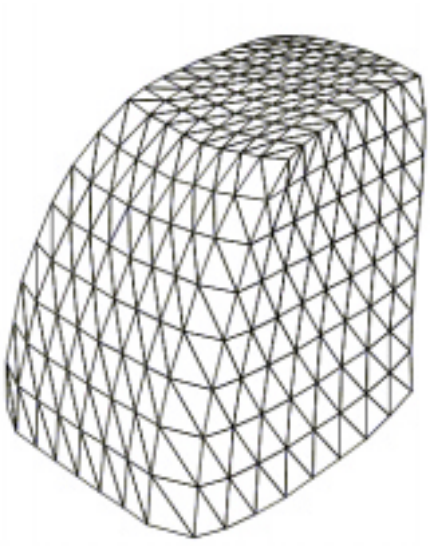
(a)



(b)



(c)



(d)

図 4.3: スピーカーのスケッチへの適用 (1)

(a): CrossPad 上で描画したスケッチ図を計算機に読み込み表示したもの
(b): 入力されたスケッチ図から抽出された芯線を表示したもの
(c): 3 次元形状を決定する稜線を結ぶ頂点と単位立方体
(d): 構築した 3 次元のソリッドモデルをメッシュ表示したもの

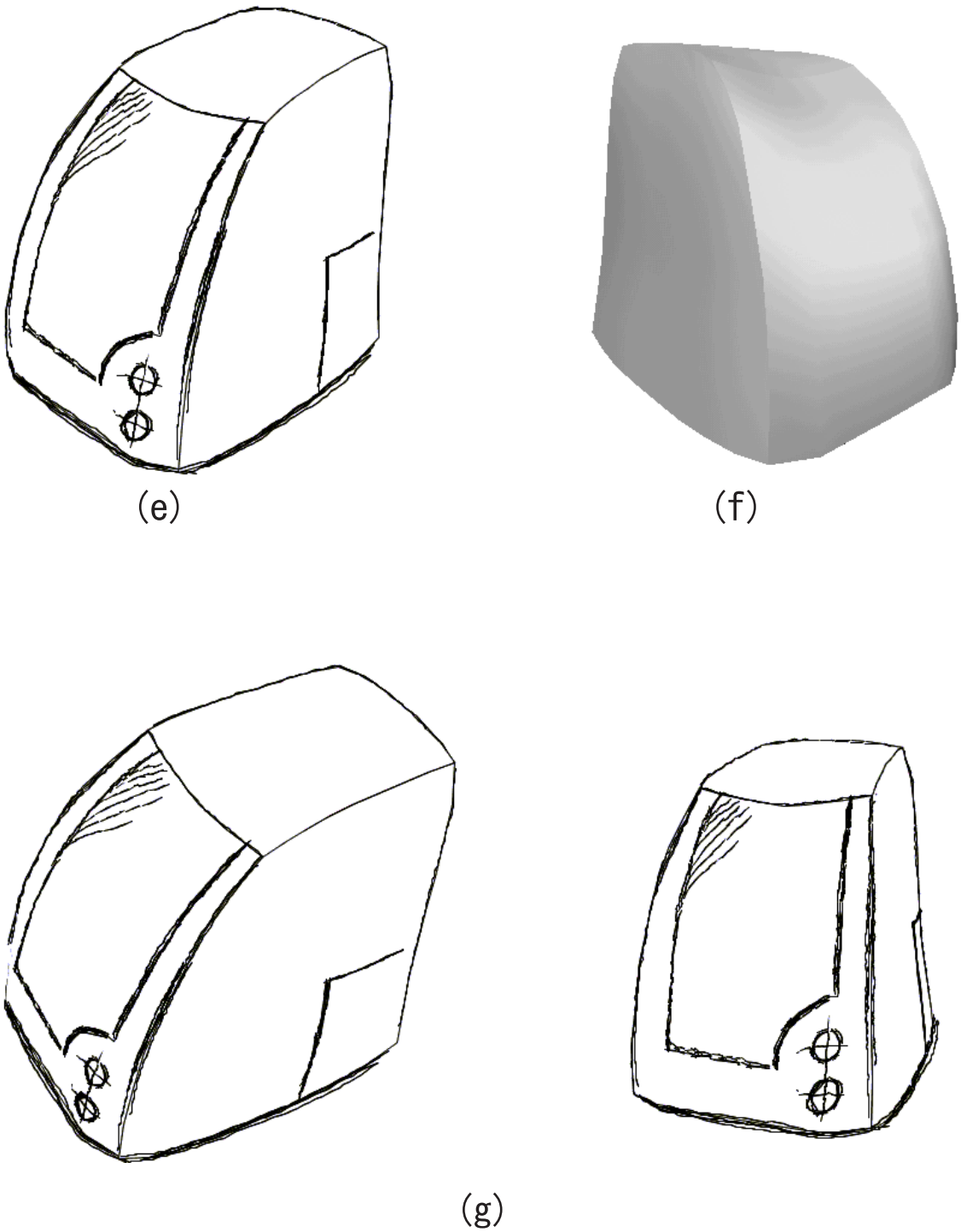


図 4.4: スピーカーのスケッチへの適用 (2)

(e): ソリッドモデルに手描きレンダリングを適用して表示したもの
(f): 得られた 3 次元形状を他の 3D 表示アプリケーションで表示したもの
(g): 得られた 3 次元形状を回転させ、別の角度から眺めたもの

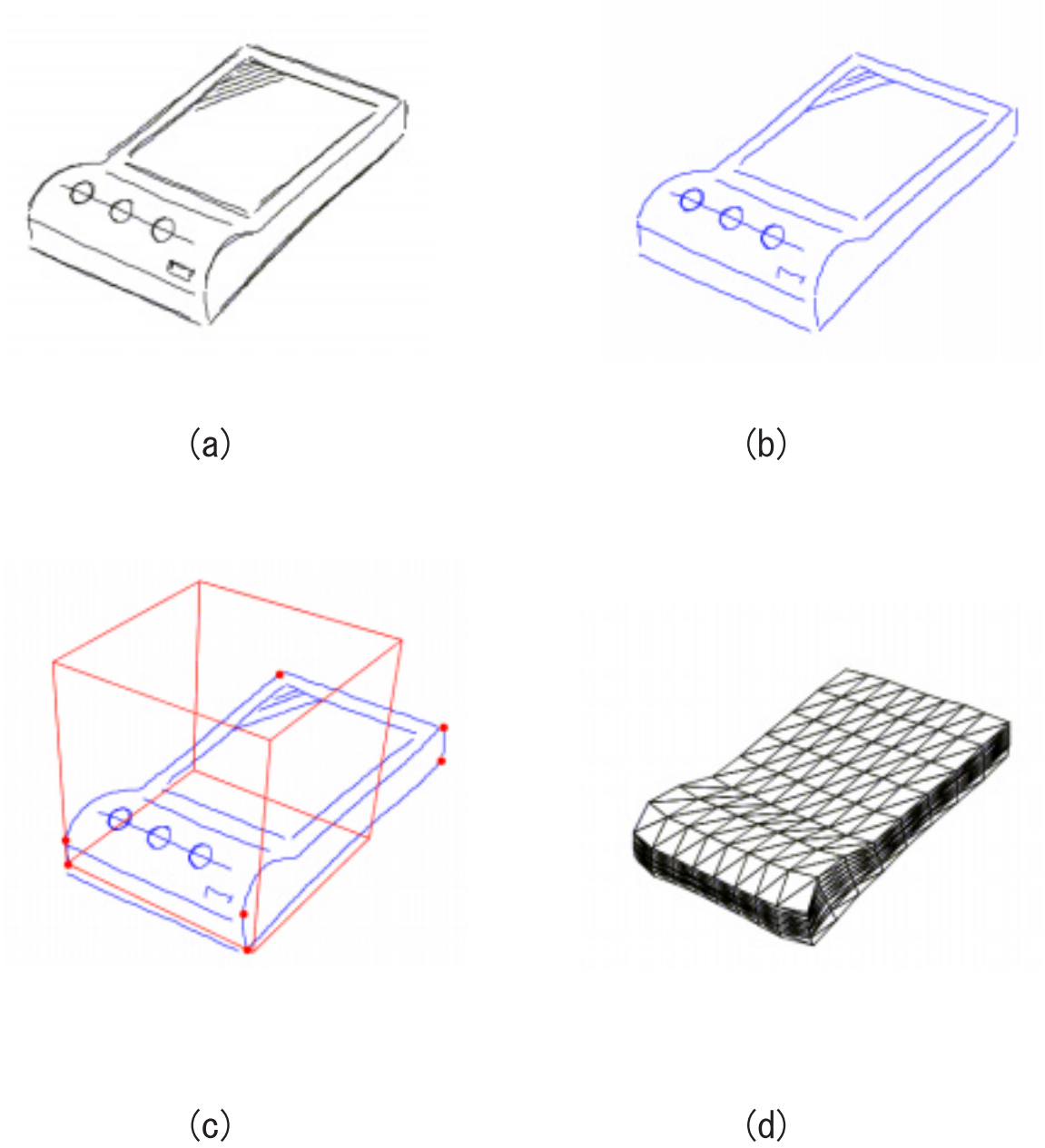


図 4.5: 情報端末のスケッチへの適用 (1)

(a): CrossPad 上で描画したスケッチ図を計算機に読み込み表示したもの
(b): 入力されたスケッチ図から抽出された芯線を表示したもの
(c): 3 次元形状を決定する稜線を結ぶ頂点と単位立方体
(d): 構築した 3 次元のソリッドモデルをメッシュ表示したもの

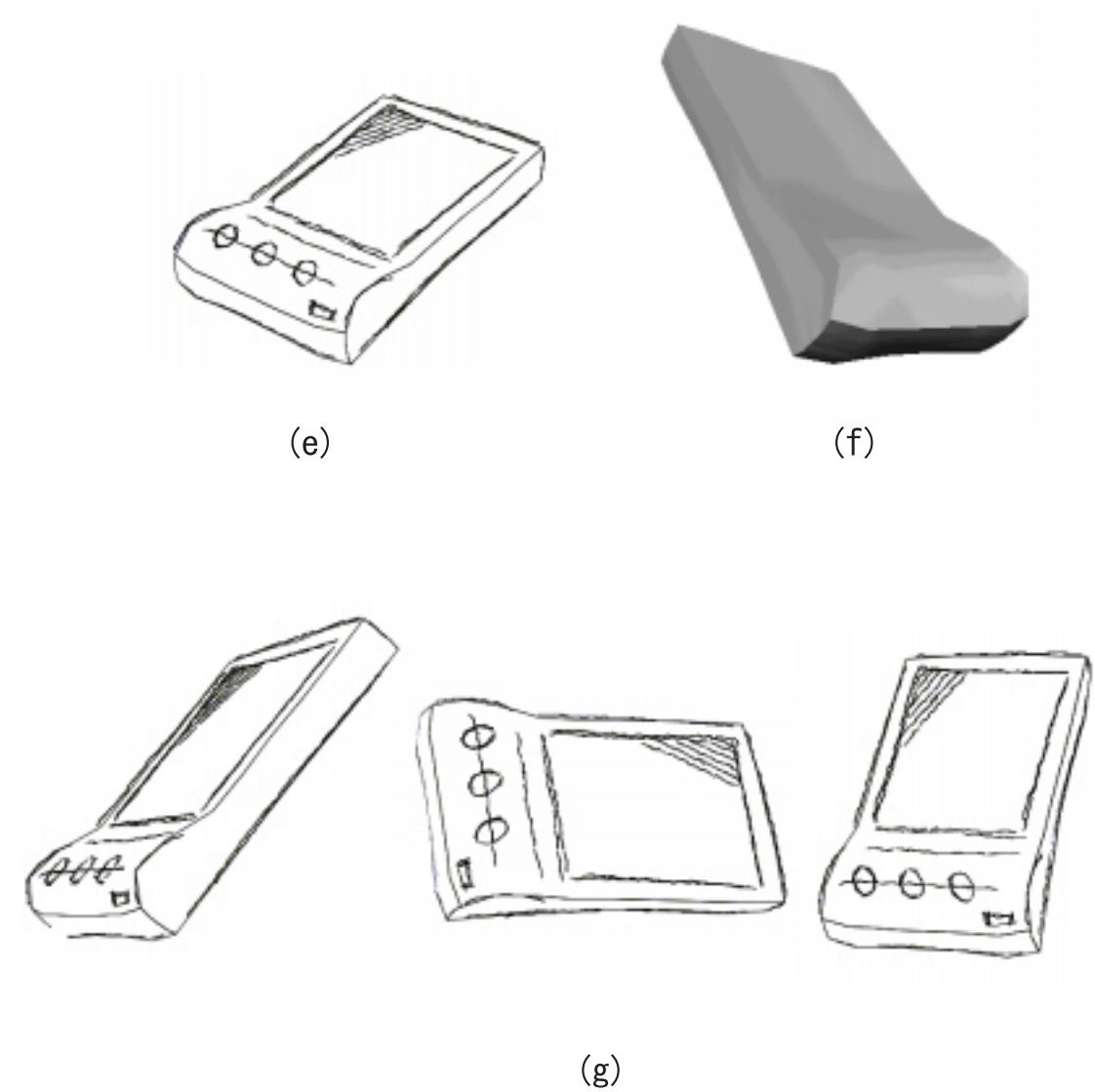


図 4.6: 情報端末のスケッチへの適用 (2)

(e): ソリッドモデルに手描きレンダリングを適用して表示したもの
(f): 得られた 3 次元形状を他の 3D 表示アプリケーションで表示したもの
(g): 得られた 3 次元形状を回転させ、別の角度から眺めたもの

4.1.3 手描きレンダリングの応用

本手法で提案した手描きレンダリングでは、デザイナのスケッチスタイルを反映したモデルの表示が行われる。

従来は計算機で表示されたモデルと実際にデザイナが描くモデルとでは、大きな違いがあったが、本手法で提案した手描きレンダリングを用いることで、得られた 3 次元形状を別の角度から投影しても、実際のスケッチと違和感のない描画が実現できる。

ここでは、発想支援の一つの試みとして、スケッチ入力から得られた 3 次元モデルを本手法で提案する手描きレンダリングで表示し、それに手描きのスケッチを追加することで、一つのシーンを描画することを行った。

手描きレンダリングでは、デザイナ個々のスタイルを反映する事ができるため、計算機によって出力された画像と、追加で描かれたスケッチが違和感なく共存することができる。

図 4.7(a) は、 CrossPad 上で描画したスケッチ図を計算機に読み込み、表示したもの、(b) は、構築した 3 次元のソリッドモデルをメッシュ表示したもの、 (c) は、得られた 3 次元形状を他の 3D 表示アプリケーションで表示したもの、 (d) は、得られた 3 次元形状を回転させ、別の角度から眺めたもの、 (e) は、別の角度から眺めたものに、液晶タブレットでスケッチを追加したものである。追加されたスケッチ自体は 3 次元情報を持っていない。

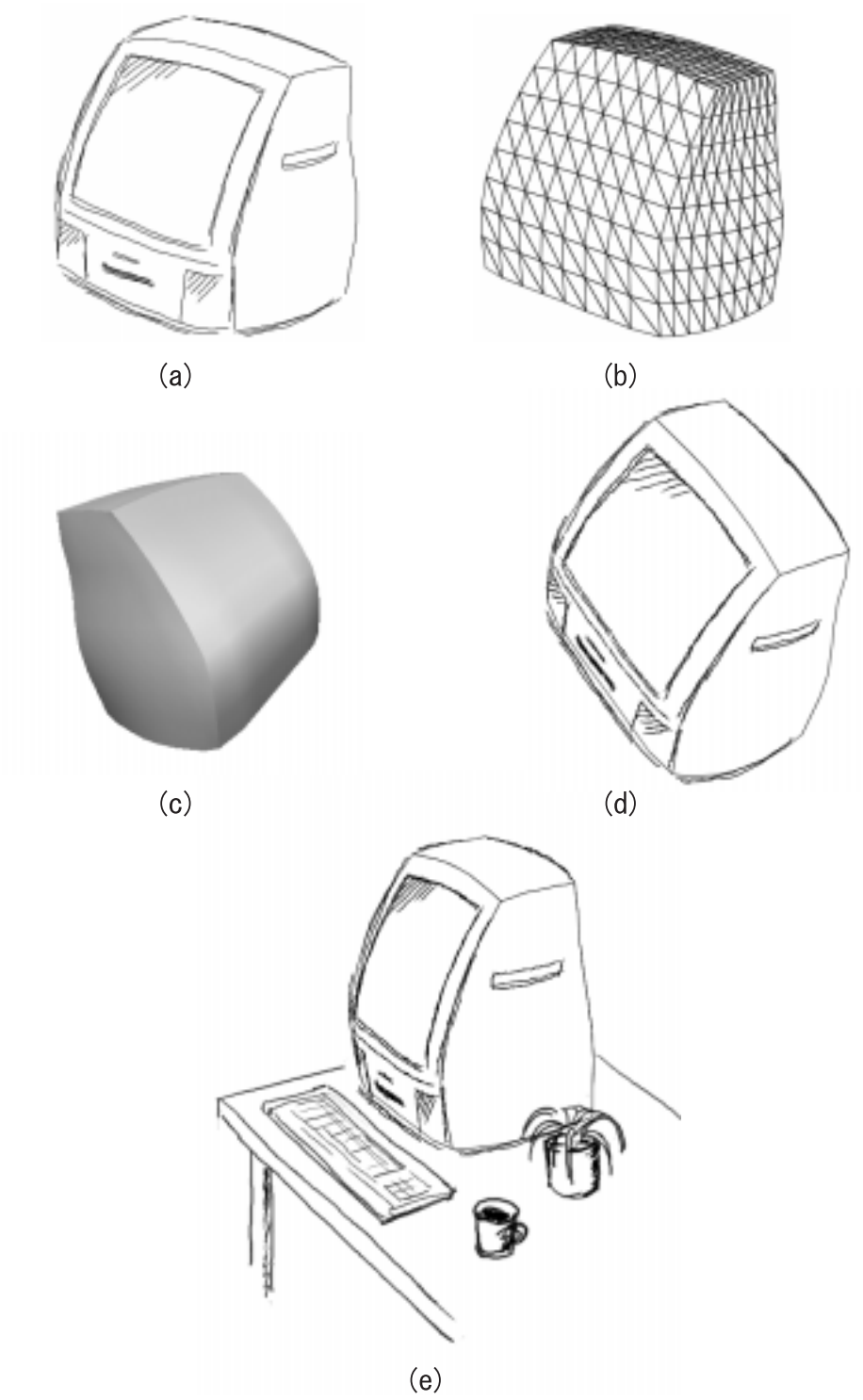


図 4.7: パーソナルコンピュータのスケッチへの適用

(a): CrossPad 上で描画したスケッチ図を計算機に読み込み表示したもの
(b): 構築した 3 次元のソリッドモデルをメッシュ表示したもの
(c): 得られた 3 次元形状を他の 3D 表示アプリケーションで表示したもの
(d): 得られた 3 次元形状を回転させ、別の角度から眺めたもの
(e): 別の角度から眺めたものに、液晶タブレットでスケッチを追加したもの

4.2 3 次元スケッチシステムの評価

3 次元スケッチを実装したアプリケーションを用い、デザイナの前でデモンストレーションを行い、その後で 2、3 の形状を実際にスケッチ図として入力してもらった。ここでは、デモンストレーションに対するデザイナの意見を中心に、本研究で提案する 3 次元スケッチの評価をまとめる。

4.2.1 インターフェースについて

スケッチ図を計算機に取り込み、自動的に 3 次元形状を構築するインターフェースについての意見として、次のようなものがあった。

- CAD の複雑な操作と比較すると遥かに楽に形状を入力できるのがよい
- スケッチで描いた形状を回転できるのがよい
- 3 次元化する手間が小さければ小さい程、作った形状を破棄する抵抗が少なく、新しい形状の作成にすぐに移行できるため、簡単に 3 次元モデルを構築できるのはよいインターフェースである
- 大きなモデルを作る場合には、2 次元のスケッチではなく、直接 3 次元のインターフェースでモデルを構築できるとよい
- CrossPad、液晶タブレットを用いても、普段使い慣れている画材と違うため、若干の抵抗がある

4.2.2 入力可能な形状について

本手法では、左右対称な六面体形状のみ入力が可能であるという拘束がある。これについての意見として、次のようなものがあった。

- 左右対称な六面体という拘束は厳しいので、任意の形状を入力できるようにしたい
- 六面体に限定した場合でも、角の無い形状 (図 4.8) を入力できるとよい
- 小さい部分の作り込みもできるとよい

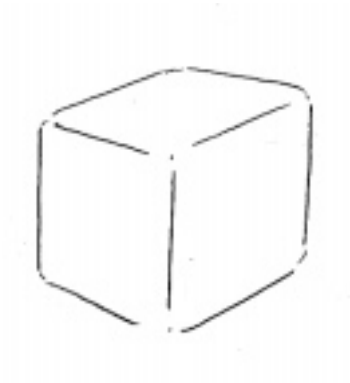


図 4.8: 角の無い立方体形状

4.2.3 手描きレンダリングについて

本手法で提案した手描きレンダリングについての意見は次のようなものがあった。

- 一般的な CG のレンダリングを行うと、そこで思考が止まってしまうので、手描きのような表示を行うのはよい
- 工業製品の表示としてだけではなく、アニメなどの娯楽向けにも使えるかもしれない
- 発想の段階で製品の材質まで考慮する時には、リアルなレンダリングの方が役立つこともある

4.2.4 その他

その他の意見には次のようなものがあった。

- 形状を入力後も、簡単なインターフェースで加筆修正を行えるといい
- 直線のつもりでデザイナが描いた線については直線として出力して欲しい
- 口で説明するのが困難な形状も簡単に構築できるようになれば、コミュニケーションツールとしても使える

第 5 章

結論と展望

第 5 章

結論と展望

本章では、本研究で得られた結論を整理し、これからの展望について述べる。

5.1　結論

本論文では、既存の CAD システムと比べてスケッチの優れている点について、形状構築段階と形状表示段階という 2 つのステージについてまとめた。これをもとに、計算機を用いて意匠設計段階の発想支援、および 3 次元形状構築の支援を行う際に重視するべき点について、実際のスケッチと同等のインターフェースによってモデルを構築できることと、デザイナのストロークを活かした形状表示ができることの 2 点に主眼を置くこととし、新しい 3 次元スケッチ手法を提案した。

また実際にシステムを実装し、評価を行うことで、以下の結論を得た。

- 本研究で提案した 3 次元スケッチ手法により、左右対称な六面体形状であるという条件のもと、単一のスケッチの入力から 3 次元形状を構築できることを示した。また、デザイナによって描かれたストローク形状を活かしたモデル表示を行う手描きレンダリングの手法により、入力のスケッチ図と同じようなモデルの表示を実現できることを示した。

- 本研究で提案した 3 次元スケッチをプログラムに実装することで、実際にスケッチの入力からモデルの構築、および手描きレンダリングまでを一貫したシステムで行えることを示した。また、モデルの構築は数十秒程度で行え、手描きレンダリングはリアルタイムでモデルを回転させることができる速度で行うことができた。

- 実際にデザイナの前でシステムのデモンストレーションを行い、その有効性を認めてもらうと共に、デザイナ側からの要望や、今後の展望などの意見を聞くことができた。

5.2 展望

本研究のさらなる展望として、以下のような拡張が考えられる。

- 本研究で提案した手法では、左右対称な六面体形状しか構築できないため、入力可能な形状の種類を増やすことが、まず第一に考えられる。似たような左右対称形状、またはプリミティブ形状であれば、本手法と同じ枠組でバリエーションをを増やすことが可能であるが、任意の形状を入力できるようにするためには、3 次元モデル構築のアルゴリズムを新たに考案する必要があるであろう。

- 本研究で提案した形状生成手法では、面を囲む境界線から Coons Patch を生成しているが、実際にはデザイナが入力した陰影線や、断面線などが考慮された曲面が生成されることが望まれる。また、曲面生成後も、スケッチと同等のインターフェースで形状修正などが可能なようにすることが望まれる。

- 本研究で作成したシステムでは、単一のモデルのみ生成可能であるが、複数の形状を生成し、それらを組み合わせることができるようになれば、より複雑なモデルも構築できると考えられる。

- 現在のシステムでは、スケッチ図の解析に失敗した場合に、そこで処理が止まってしまうが、途中でユーザの介在を許すなどして、より柔軟なシステムにすることも考えられる。

謝辞

東京大学大学院工学系研究科精密機械工学専攻、鈴木宏正助教授には、本研究の直接の指導教官として、研究テーマおよび方針をはじめ、多くの貴重な御指導を頂きました。私の質問に、いつも丁寧に答えてくださりました。ここに深く感謝の意を表わします。

東京大学大学院工学系研究科精密機械工学専攻、木村文彦教授には、本研究の指導教官として、研究の方針や問題点について多くの貴重な御指導を頂きました。ここに深く感謝の意を表します。

東京大学大学院工学系研究科精密機械工学専攻、碇山みちこ助手には、 研究室で生活する上で大変お世話になりました。ここに深く感謝の意を表します。

東京大学大学院工学系研究科精密機械工学専攻博士課程 2 年、川地克明氏には計算機の使用法について、様々なことを教えて頂きました。常に多忙であったにも関わらず、快く御時間を割いて頂き、あらゆる質問に答えて下さいました。ここに深く感謝の意を表します。

東京大学大学院工学系研究科精密機械工学専攻修士課程 1 年、秦智之氏には、計算機の使用について幾多の助言をいただきました。ここに深く感謝の意を表します。

東京大学大学院工学系研究科精密機械工学専攻修士課程 2 年、坂本裕和氏、田中英人氏、野中亮吾氏には、共に修論研究に取り組む同士としてだけでなく、研究生活全般において、公私に渡り大変お世話になりました。ここに深く感謝の意を表します。

東京大学工学部精密機械工学科木村・鈴木研究室の方々には、いろいろな面で援助して頂きました。ここに深く感謝の意を表します。

最後に、いつも私を応援してくれている家族に感謝します。

参考文献

[Akeo94] M. Akeo, H. Hashimoto, T. Kobayashi, and T. Shibusawa: Computer Graphics system for reproducing three-dimensional shape from idea sketch, In *Eurographics '94 Proceedings*, volume 13, pp. 477–488 (1994).

[Baudel94] T. Baudel: A Mark-Based Interaction Paradigm for Free-Hand Drawing, In *in Proc. of UIST '94*, pp. 185–192 (1994).

[Branco94] V. Branco, A. Costa, and F. N. Ferriera: Sketching 3D models with 2D interaction devices, In *in Proc. of Eurographics '94*, volume 13, pp. 489–502 (1994).

[Chen96] C. P. Chen and S. Xie: Freehand drawing system using a fuzzy logic concept, In *Computer-Aided Design*, volume 28, pp. 88–89 (1996).

[Curless96] B. Curless and M. Levoy: A Volumetric Method for Building Complex Models from Range Images, In *Proceedings of ACM SIGGRAPH '96*, pp. 303–312 (1996).

[C.Zeleznik96] R. C.Zeleznik, K. P.Herndon, and J. F.Hughes: SKETCH:An Interface for Sketching 3D Scene, In *Proceedings of ACM SIGGRAPH '96*, pp. 163–170, ACM Press (1996).

[Debevec96] P. E. Debevec, C. J. Taylor, and J. Malik: Modeling and Rendering Architecture from Photographs: A hybrid geometry- and image-based approach, In *Proceedings of ACM SIGGRAPH '96*, pp. 11–20 (1996).

[Eck96] M. Eck and H. Hoppe: Automatic Reconstruction of B-Spline Surfaces of Arbitrary Topological Type, In *Pro-*

ceedings of ACM SIGGRAPH '96, pp. 325–334, ACM Press (1996).

[Farin97] G. Farin: *Curves and Surfaces for Computer-Aided Geometric Design*, ACADEMIC PRESS (1997).

[Gooch98] A. Gooch, B. Gooch, P. Shirley, and E. Cohen: A Non-Photorealistic Lighting Model For Automatic Technical Illustration, In *Proceedings of ACM SIGGRAPH '98*, pp. 447–452 (1998).

[Han97] S. Han and G. Medioni: 3DSketch: Modeling by Digitizing with a Smart 3D Pen, In *in Proc. of ACM Multimedia '97*, pp. 41–48 (1997).

[IBM] IBM: CrossPad, http://www.jp.ibm.com/pc/vlp/vhcrp8b/vhcrp8ba.htm

[Igarashi99] T. Igarashi, S. Matsuoka, and H. Tanaka: Teddy:A Sketching Interface for 3D Freeform Design, In *Proceedings of ACM SIGGRAPH '99*, pp. 409–416, ACM Press (1999).

[Keller97] A. Keller: Instant Radiosity, In *Proceedings of ACM SIGGRAPH '97*, pp. 49–56 (1997).

[Markosian97] L. Markosian, M. A. Kowawlski, S. J. Trychin, L. D. Bourdev, D. Goldstein, and J. F.Hughes: Real-Time Nonphotorealistic Rendering, In *Proceedings of ACM SIGGRAPH '97*, pp. 415–420, ACM Press (1997).

[Pentland89] A. Pentland, J. Kuo, and M. M. Lab: The Artist at the Interface, In *Vision Science Technical Report 114* (1989).

[P.H. ウィンストン 79] P.H. ウィンストン, 白井良明, 杉原厚吉: コンピュータービジョンの心理, 産業図書 (1979).

[Rademacher99] P. Rademacher: View-Dependent Geometry, In *Proceedings of ACM SIGGRAPH '99*, pp. 439–446 (1999).

[Schkolne99] S. Schkolne and P. Schroder: Surface Drawing, In *Caltech Computer Science CS-TR-99-0* (1999).

[Sugishita96] S. Sugishita, K. Kondo, H. Sato, and S. Shimada: Sketch Interpreter for geometric modelling, In *Annals of Numerical Mathematics 3*, pp. 361–372 (1996).

[Tanaka89] T. Tanaka, S. Naito, and T. Takahashi: Generalized symmetry and its application to 3D shape generation, In *Visual Computer*, pp. 83–94 (1989).

[Trucco98] E. Trucco and A. Verri: *INTRODUCTORY TECHNIQUES for 3-D COMPUTER VISION*, Prentice-Hall, Inc. (1998).

[WACOM] WACOM: WACOM, http://tablet.wacom.co.jp/.

[(株) エヌケー・エクサ] (株) エヌケー・エクサ: Design Spinnake, http://www.nk-exa.co.jp/ds/.

[機械システム振興協会 99] 機械システム振興協会: コンテンツ制作支援システムに関する調査研究報告書, Technical report, 機械システム振興協会 (1999).

[近藤 88] 近藤邦雄, 木村文彦, 田嶋太郎: 手描き透視図の視点推定とその応用, 情報処理学会論文誌, volume 29, pp. 686–693 (1988).

[古島] 古島終作: 六角大王, http://plaza10.mbn.or.jp/ shusaku/roku/roku.html

[古島 93] 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, pp. 105–110 (1993).

[視覚デザイン研究所編集室 94] 視覚デザイン研究所編集室: 鉛筆画ノート, 視覚デザイン研究所 (1994).

[松田 99] 松田浩一, 近藤邦雄: 手書き図形入力のための時系列情報を利用した逐次清書法, 情報処理学会論文誌, volume 40, pp. 594–601 (1999).

[清水吉治 98] 清水吉治, 小島孝, 田野雅三, 松田真次: モデリングテクニック, グラフィック社 (1998).

[西野 99] 西野浩明, 凍田和美, 椛田泰行, DahlanNariman, 宇都宮孝一: 両手ジェスチャによる 3 次元物体の生成と変形, 情報処理学会論文誌, volume 20 (1999).

[千葉則茂 97] 千葉則茂, 土井章男: 3 次元 CG の基礎と応用, サイエンス社 (1997).