〔非公開〕

TR－C－0026

光切断法による
3次元形状の自動入力

西野　治彦　HARUHIKO NISHINO
秋山　健二　KENJI AKIYAMA
小林　幸雄　YUKIO KOBAYASHI

1989. 2. 28.

ＡＴＲ通信システム研究所

目　次

1. はじめに

本報告は、1987年3月から1989年2月までの2年間、(株)ATR通信システム研究所知能処理研究室において、光切断法による3次元形状の自動入力について行った検討結果をまとめたものである。

3次元コンピュータグラフィックス画像、あるいは、立体アニメーション等を作成するには、個々の物体の形状・テクスチャ・材質等のデータを登録した3次元物体データベースを構築する必要がある。現在、実際の物体の形状データを入力するには、3次元ディジタイザ等を使用して、物体表面上の一点一点の座標を人手で登録する方法が一般的であり、多大な時間・労力を要しているのが実情である。したがって、こうした物体形状データの入力作業の自動化が強く望まれているが、現在のところ、複雑な形状の物体に対しても十分満足できる自動入力装置は、実現されていない。一方、コンピュータビジョンやロボティクス等の分野においては、非接触な手法で距離情報を計測するさまざまな方法が開発されている[5][7][13][29]。これらは、ステレオ視のような受動的な方法と、光切断法・モアレ法[9]・飛行時間測定法のように物体に対して光等を投げかける能動的な方法に分類できる。ステレオ視は、画素間の対応付けに決定的な方法がなく、現在のところ、形状入力のような用途には能動的な方法が有力である。

本研究では、能動的な距離計測法の一種である光切断法を用いた物体形状の自動入力方法、特に、計測したデータからの立体形状の再構成方法、物体表面の平面の抽出方法を検討する。光切断法による3次元計測は、従来からの技術で、今日では産業界でも利用されており、比較的簡単な方法で、かつ、高精度に3次元座標を計測できる[16][19]。本研究で行った光切断法による形状入力方法は、回転テーブルを利用して物体を回転させながら、物体の全周囲の形状を入力し、三角パッチの集合で立体形状を再構成を行うものである。回転テーブルを用いて、非接触な計測方法で物体の全周囲の形状データを計測しようとするアプローチは、これまでにも行われている[2][25][28]が、計測データをもとに立体形状をどう表現すればよいかを検討した例は少ない。そのうちSchmittら[25]は、計測データをもとに大きな三角パッチを徐々に分割していく表現方法を提案している。本研究では、光切断法で計測したデータをもとに、立体形状を三角パッチの集合で再構成することとし、その節点の発生方法[18][20]、回転テーブルで物体を回転させるだけでは欠落する形状を補完して、完全な立体形状データを生成する方法[21][23]等の再構成方法を検討する。また、もう一つの研究テーマとして、光切断法で計測したデータをもとに、物体表面の平面抽出方法[22]を検討する。物体表面の平面が認識できれば、その平面情報を用いて、物体認識や形状の表現方

法に応用が可能である。本研究では、平面の抽出方法として、３次元Ｈｏｕｇｈ変換による方法を検討した。従来、３次元空間上の計測点の集合から３次元Ｈｏｕｇｈ変換で平面を抽出する方法には、膨大な計算量が必要であった。本研究では、光切断法を用いれば、物体表面上の直線データを収集できることに着目し、直線データを直接３次元Ｈｏｕｇｈ変換することによる平面抽出方法を検討した。

本文では、以上のテーマについて、３章に分けて検討結果を述べる。

まず、第２章では、光切断法による形状計測システムと三角パッチによる再構成方法について述べる。回転テーブル上の物体をわずかな角度回転させるごとに光切断法で計測することにより、全周囲の形状を入力する。回転テーブルの各回転角度で計測したデータを折れ線で近似し、各線分の端点から三角パッチの節点を発生させ、隣接する回転角度で計測したデータ上の節点相互間に三角パッチを生成する。このように、第２章では、回転テーブル上の物体について、その全周囲の形状の計測、及び、その形状データの再構成方法を述べる。

次に、第３章では、回転テーブルで物体を回転させるだけでは計測できない部分の形状を補完して、欠落部分のない完全な形状データを生成する方法を述べる。第２章で述べた方法により、全周囲の形状の入力が可能であるが、上部・下部等のスリット光の当り方が弱かった部分、及び、ＴＶカメラから死角となった部分では形状データが欠落する。この欠落した部分の形状を、回転テーブル上への物体の置き方をかえて入力した形状データで補完し、欠落部分のない完全な形状データの生成方法を述べる。

次に第４章では、光切断法で計測したデータから物体表面の直線部分を検出し、その直線データを直接３次元Ｈｏｕｇｈ変換する（直線型３次元Ｈｏｕｇｈ変換と呼ぶ）ことにより、物体表面上の平面を抽出する方法を述べる。

最後に第５章で、全体のまとめと今後の課題を述べる。

２．形状計測システムと形状再構成方法

２．１　形状計測システム

２．１．１　計測原理

光切断法による形状計測システムでは、図２．１に示すように、回転テーブル上に入力対象物体を置き、スリット光を回転テーブルの回転軸に向けて投影する。ＴＶカメラで入力した画像から、ディジタル画像処理により、スリット光が投影された部分（以下、画像上におけるスリット光投影部分を変形スリット像と呼ぶ）を抽出し、三角測量の原理で物体表面上のスリット光投影部分の３次元座標を求める。全周囲方向からの形状データを計測するため、回転テーブルを利用して入力対象物体をわずかな角度（$\Delta\theta$）だけ回転させるごとにこの計測を行う。本計測方法では、スリット光を回転テーブルの回転軸方向に向けているため、回転テーブルの回転軸をｚ軸とする円柱座標系（ｒ，θ，ｚ）を用いれば、各々の計測データは、ある偏角θの方向の形状データとなる。

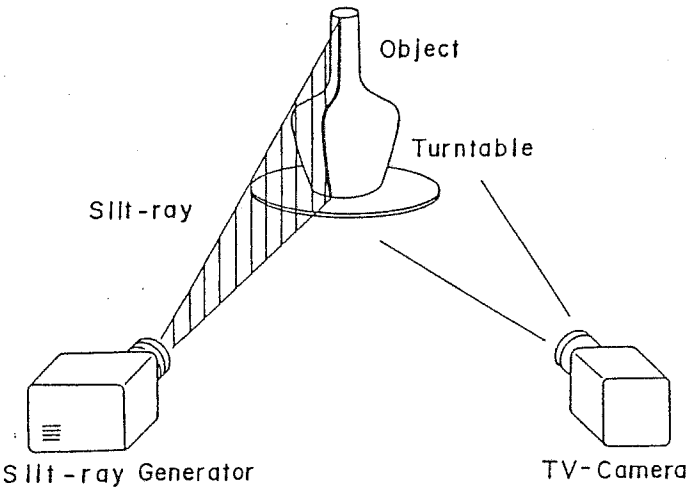


図２．１　計測原理

２．１．２　計測システムの構成

本研究のため構築した形状計測システムのハードウェア構成を図２．２に示す。スリット光は、高出力スライドプロジェクタを使用して、特殊加工したガラススライドを投影することにより発生させた。ＶＩＣＯＭは、ＳＵＮ－３／２６０Ｃワークステーション機能に、画像処理機能が付加された汎用画像処理ワークステーションである。ＶＩＣＯＭでは、画像の取り込み（8bit量子化）、画像上の変形スリット像の抽出、３次元座標への変換、三角パッチによるサーフェイスモデルの再構成等の処理を行う。画像処理は、横 512×縦 476 画素のサイズで行った。

回転テーブルは、ＶＩＣＯＭとＲＳ－２３２Ｃケーブルで接続され、ＶＩＣＯＭからの信号により任意の角度に制御可能である。ＩＲＩＳグラフィックワークステーションでは、任意の角度からのワイヤフレームによる表示、レンダリング等の３次元コンピュータグラフィックス処理が可能である。

図２．３は、回転テーブル上に置かれた入力対象物体（ビーナス像）であり、図２．４は、ビーナス像にスリット光を投影したところである。このような画像から変形スリット像部分を抽出し、スリット光投影部分の３次元座標を求める。

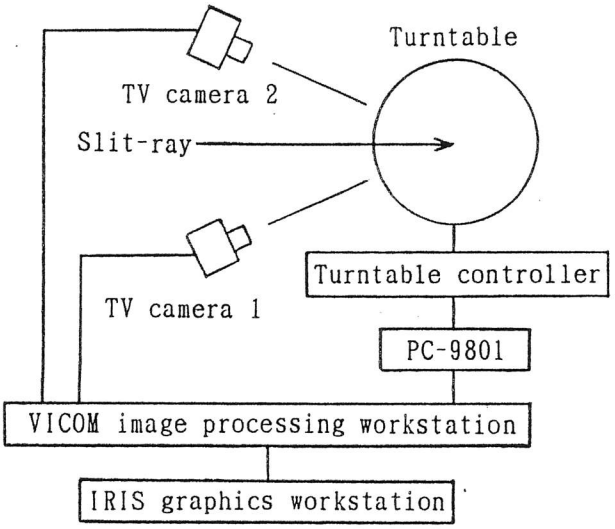


図２．２　計測システムの構成

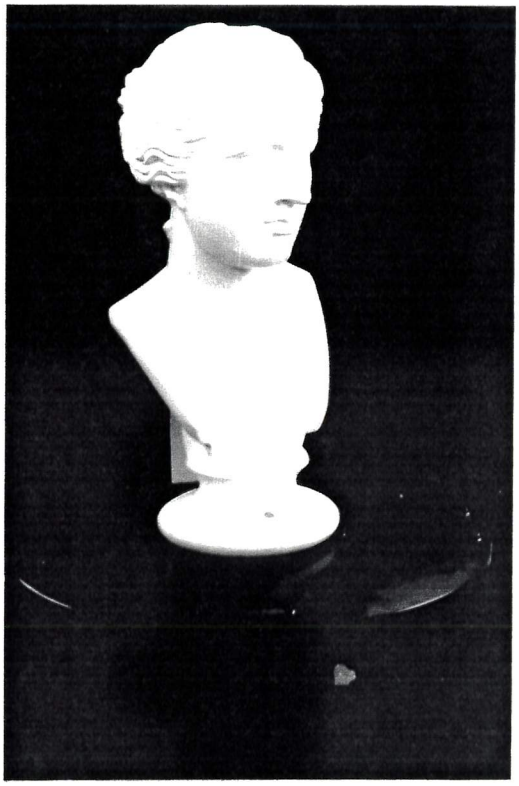


図２．３　回転テーブル上の入力対象物体

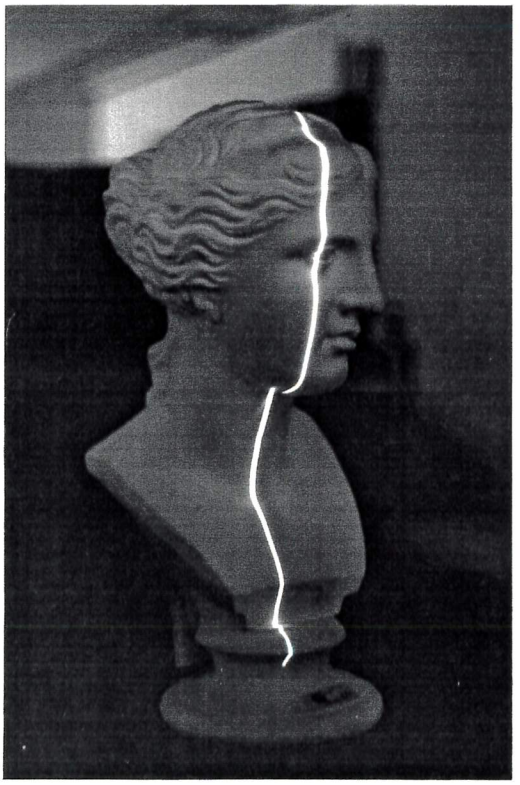


図２．４　スリット光の投影

2．1．3　変形スリット像の抽出

ＴＶカメラで入力された画像は、サンプリング・量子化処理され、画像メモリに取り込まれる。画像メモリの座標を（ｕ，ｖ）で表わし、画素（ｕ，ｖ）の輝度をＩ（ｕ，ｖ）で表現する。ＴＶカメラをほぼ水平に設置しているため、画像上の変形スリット像は、物体表面の凹凸に応じて水平方向（ｕ座標方向）に変化する。したがって、変形スリット像を抽出するのに各ｖ座標の値ごとに（各水平ラインごとに）変形スリット像のｕ座標の値を求める方法を採用した。

変形スリット像のｕ座標値を１画素以下の分解能で求める試みはこれまでにも行われている[4][16]が、これらの方法では、かなりの計算量が必要である。本計測システムでは、画素間の輝度の比を計算する簡単な方法により、1/4画素の分解能で変形スリット像の位置（u_0^*，v_0）を求めた。まず、水平ライン（v_0）で、最高輝度画素（u_0）を求める。ただし、L_{th}を閾値として、$I(u_0, v_0) \geqq L_{th}$を満足するものとする。このあと、その輝度Ｉ（u_0，v_0）とその左右画素の輝度とを比較して、変形スリット像の位置（u_0^*，v_0）を1/4画素単位で左右にシフトさせる。ここで、左画素の輝度をＩ（u_0-1，v_0）、右画素の輝度をＩ（u_0+1，v_0）で表わす。

(i)　$I(u_0-1, v_0) \geqq I(u_0+1, v_0)$ のとき

- $I(u_0-1, v_0)/I(u_0, v_0) \geqq R_{th1}$ であれば　　$u_0^* = u_0 - 0.5$
- $I(u_0-1, v_0)/I(u_0, v_0) < R_{th1}$ かつ、
  $(I(u_0, v_0)-I(u_0-1, v_0))/(I(u_0, v_0)-I(u_0+1, v_0)) \geqq R_{th2}$ であれば、
  $u_0^* = u_0$
- $I(u_0-1, v_0)/I(u_0, v_0) < R_{th1}$ かつ、
  $(I(u_0, v_0)-I(u_0-1, v_0))/(I(u_0, v_0)-I(u_0+1, v_0)) < R_{th2}$ であれば、
  $u_0^* = u_0 - 0.25$

(ii) $I(u_0-1, v_0) < I(u_0+1, v_0)$のとき

- $I(u_0+1, v_0)/I(u_0, v_0) \geqq R_{th1}$であれば、$u_0^* = u_0 + 0.5$
- $I(u_0+1, v_0)/I(u_0, v_0) < R_{th1}$ かつ、
  $(I(u_0, v_0)-I(u_0+1, v_0))/(I(u_0, v_0)-I(u_0-1, v_0)) \geqq R_{th2}$ であれば、
  $u_0^* = u_0$
- $I(u_0+1, v_0)/I(u_0, v_0) < R_{th1}$ かつ、
  $(I(u_0, v_0)-I(u_0+1, v_0))/(I(u_0, v_0)-I(u_0-1, v_0)) < R_{th2}$ であれば、
  $u_0^* = u_0 + 0.25$

ここで、R_{th1}、R_{th2}は、変形スリット像位置のシフト量を決定する閾値であり、この値により、変形スリット像の座標値u_0'は、整数値、整数値＋0.25、整数値＋0.5、整数値＋0.75 の４種類に分類される。本来、この４種類に分類される割合は、統計的に25%ずつになるはずであるが、そのための閾値R_{th1}、R_{th2}の値は、入力対象物体の材質、ＴＶカメラの調整等に応じて変化する。そこで、最初に入力した１画面の中で変形スリット像部分の輝度分布の統計をとり、そのデータから各25%ずつに分類されるように閾値R_{th1}、R_{th2}の値を決定した。

この手法は、1/8画素単位、1/16画素単位…に容易に拡張可能である。図２．５は、本形状計測システムにおいてこの多値画像処理の効果を実験した結果であるが、1/8画素以下の検出分解能にしても、画像信号に含まれるノイズ成分の影響等により、計測精度の大きな改善効果は期待できない。そのため、本計測システムでは1/4画素単位の処理を行っている。

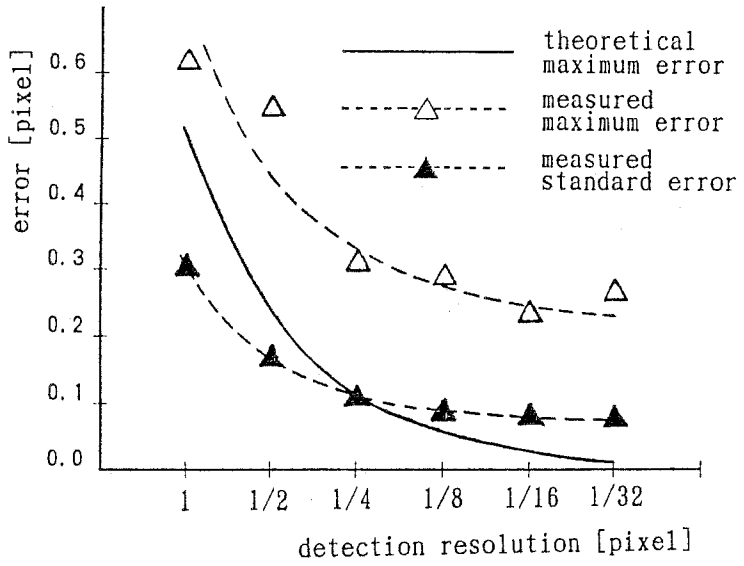


図２．５　多値画像処理による計測精度の向上

２．１．４　３次元座標への変換

画像上の座標（ｕ，ｖ）に対応する３次元空間上の座標（ｘ，ｙ，ｚ）を求めるには、ＳＰＩＤＥＲIIの変換マトリクスによる方法[8]を使用した。

この方法では、まず、３次元空間上の６個以上の既知座標と、それらに対応するカメラ入力画像上での座標の関係から、最小２乗法でカメラ変換マトリクスＡを求める。このマトリクスＡは、３次元空間上の任意の点の座標（ｘ，ｙ，ｚ）をカメラ画像上の座標（ｕ，ｖ）に変換する。

$$[s\,u,\ s\,v,\ s]^t = A\,[x,\ y,\ z,\ 1]^t \qquad (2.1)$$

次に、カメラ変換マトリクスAとスリット光平面の方程式から、光切断法のセンサーマトリクスBを求める。その原理は、カメラ画像上の座標（u，v）に対する視線とスリット光平面の交点を計算するものである。このセンサーマトリクスBは、画像上の座標（u，v）をスリット光平面上の3次元座標（x，y，z）に変換する。

$$[s\,x,\ s\,y,\ s\,z,\ s]^t = B\,[u,\ v,\ 1]^t \qquad (2.2)$$

このセンサーマトリクスBを使って、画像上の変形スリット像の座標から物体表面上の3次元座標に容易に変換することが可能である。

カメラ変換マトリクスAを求めるには、3次元空間上の6点以上の既知座標とそれに対応するカメラ画像上の各座標が必要である。図2.6に示す板を回転テーブル上に設置し、この板の上の数カ所にマークを貼り（図2.7参照）、既知の基準座標とした。さらに、板を回転させるごとにカメラ入力した画像上のマークの座標をもとにして、計測系のキャリブレーションを行った。画像上でのマーク座標の検出は、人手の入力で行ったが、カメラ画像からマークの抽出が自動的にできれば、キャリブレーションの完全自動化が可能になる。

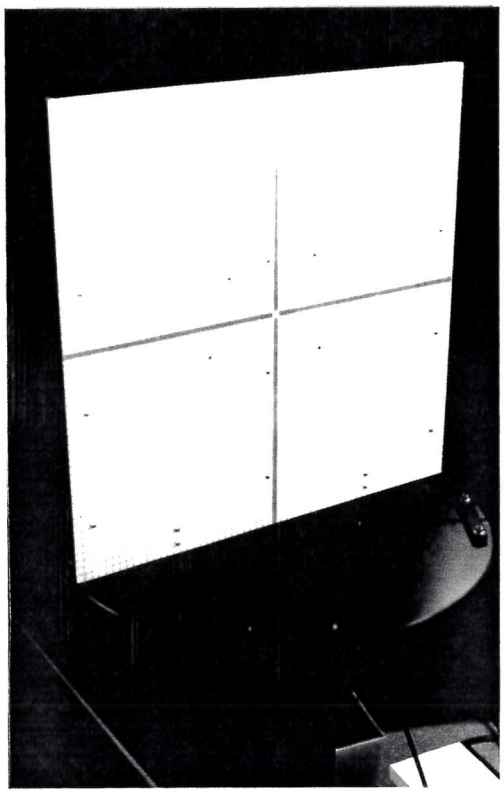


図2.6　キャリブレーション板

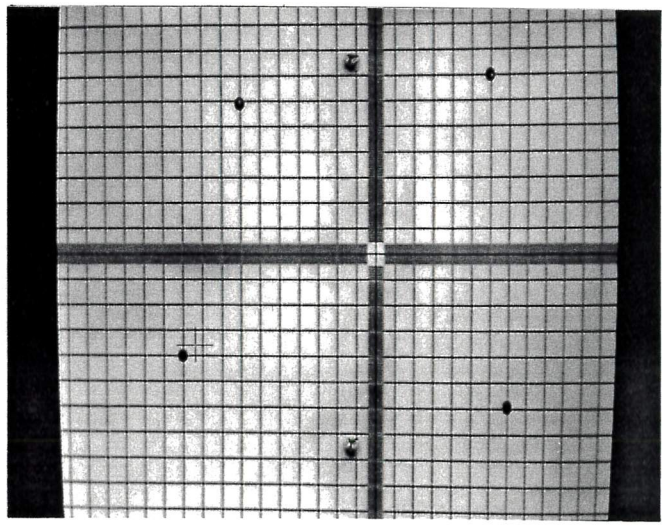


図2.7　板上のマーク

２．１．５　２台のカメラによる死角の改善

光切断法の計測精度を向上させるには、スリット光とカメラの光軸がなす角度を大きくするのがよい。しかし、この角度を大きくするほど、図２．８に示すように、物体表面のスリット光投影部分がカメラから観測できない場合、すなわち、死角が発生する。そこで、スリット光に対して反対側にもう１台のカメラ（ＴＶカメラ２）を設置し、もとのカメラ（ＴＶカメラ１）からは死角のため得ることができなかった座標データをＴＶカメラ２のデータで補った。光切断法ではスリット光を当てることができるすべての部分の表面座標データを得るのが最終的な目標であるが、２台のカメラを使用することで、光の当り方が弱い部分と穴・溝等を除くすべての形状データを得ることができる（通常、スリット光とカメラ光軸との角度は、約30[deg]として計測した。）。

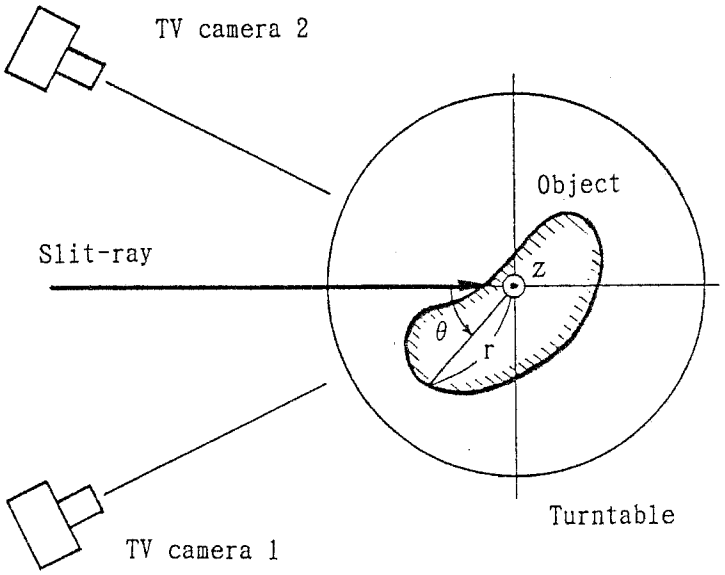


図２．８　２台のＴＶカメラによる計測システム

ＴＶカメラ１のカメラ変換マトリクスをA_1、センサーマトリクスをB_1、カメラ画像上の座標を（u，v）とし、また、ＴＶカメラ２のカメラ変換マトリクスをA_2、センサーマトリクスをB_2、カメラ画像上の座標を（u'，v'）とすれば、(2.1)(2.2)式と同様に(2.3)～(2.6)式が成立する。

$$[s_1u,\ s_1v,\ s_1]^t = A_1[x,\ y,\ z,\ 1]^t \qquad (2.3)$$

$$[s_2x,\ s_2y,\ s_2z,\ s_2]^t = B_1[u,\ v,\ 1]^t \qquad (2.4)$$

$$[s_3u',\ s_3v',\ s_3]^t = A_2[x,\ y,\ z,\ 1]^t \qquad (2.5)$$

$$[s_4x,\ s_4y,\ s_4z,\ s_4]^t = B_2[u',\ v',\ 1]^t \qquad (2.6)$$

これらの関係から、ＴＶカメラ１の画像上の座標（ｕ，ｖ）と対応するＴＶカメラ２の画像上の座標（ｕ'，ｖ'）の相互間は、次の(2.7)(2.8)式で簡単に変換することができる。

$$[s_5u',\ s_5v',\ s_5]^t = A_2B_1[u,\ v,\ 1]^t \qquad (2.7)$$

ただし、$s_5 = s_3s_2$

$$[s_6u,\ s_6v,\ s_6]^t = A_1B_2[u',\ v',\ 1]^t \qquad (2.8)$$

ただし、$s_6 = s_1s_4$

そこで、ＴＶカメラ２の画像上のデータを(2.8)式を使ってＴＶカメラ１の画像上に変換し、ＴＶカメラ１の画像上における変形スリット像を補った。

２．２　立体形状の再構成方法

２．１で述べた立体形状計測システムでは、変形スリット像を構成する各点に対して３次元座標を計算することができる。立体形状を三角パッチの集合で再構成するのに、これらのすべての点を用いれば、形状データを表現するのに必要なデータ量は、膨大なものとなってしまう。そこで、データ量を圧縮し、かつ、形状を特徴づける点を保存して、原形状との誤差を小さくするような再構成方法を検討した。本方法は、基本的には、回転テーブルの各回転角度で計測したデータを折れ線で近似し、各線分の端点から三角パッチの節点を発生させる。そして、隣接する回転角度上のデータから発生させた節点相互間を接続して、三角パッチを生成する。

２．２．１　節点の発生方法

回転テーブルを$\Delta\theta$回転させるごとに計測したデータを折れ線で近似し[1][27]、各線分の端点を三角パッチの節点とする。計測データを折れ線で近似する場合、ＴＶカメラで入力した画像上の変形スリット像の点列に対して行う方法と、画像上の変形スリット像を構成する一点一点をスリット光平面上に変換した３次元空間上の点列に対して行う方法が考えられる。図２．８に示すように、ＴＶカメラを基準にすると、スリット光は斜め後方から投影させる。このため、画像上の１画素に対応するスリット光平面上での水平方向の長さは、カメラ画像のｕ座標の値により異なる。しかし、通常使用する範囲での変化分は、最大10%程度であり、また、変形スリット像を構成するすべての点をスリット光平面上に変換する計算量も考慮して、ここでは、折れ線近似を前者の方法で行った。

折れ線近似の近似度の評価基準としては、原点列と折れ線間の距離を用いた。変形スリット像を構成する点列を$P_1 \sim P_n$とし、L_{ij}を点P_iとP_j（$1 \leqq i < j$

≦ｎ）を結ぶ線分とする。また、Ｄ（P_k，L_{ij}）（ｉ＜ｋ＜ｊ）を点P_kから線分L_{ij}に下ろした垂線の長さとする。このとき次の(2.9)式が成立すれば、点P_iとP_jの間の点列は線分L_{ij}で近似可能であると定義する（図２．９参照）。

$$\underset{i<k<j}{\mathrm{Max}}\ D(P_k, L_{ij}) \leqq D_{th} \qquad (2.9)$$

ここで、D_{th}は、折れ線近似のための閾値であり、通常、0.5～1.0[pixel]の値を用いた。本折れ線近似アルゴリズムでは、この近似条件を極大的に満足する線分の系列で、変形スリット像の点列を近似する。ここで、極大とは、線分L_{ij}で点P_i～P_j間の点列が近似可能であり、線分$L_{i(j+1)}$では、点P_i～P_{j+1}間の点列を近似できない場合をいう。

この折れ線近似アルゴリズムによって、画像上の変形スリット像を折れ線で近似し、折れ線を構成する各線分の端点の座標を２．１．４で述べた変換方法で３次元空間上に変換し、三角パッチの節点とした。

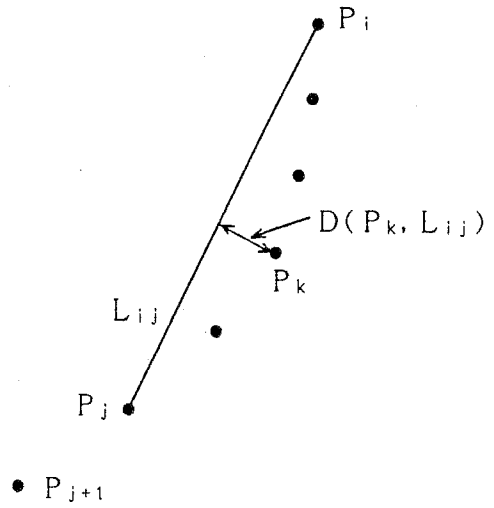


図２．９　近似可能性

２．２．２　三角パッチの生成方法

回転テーブルの回転角（θ_iとする）上の節点と隣接する回転角（θ_{i+1}とする）上の節点相互間を接続して三角パッチを生成する。ここで、θ_i上の節点系列をS_{i1}，S_{i2}，S_{i3}，…S_{im}、これに隣接するθ_{i+1}上の節点系列を$S_{(i+1)1}$，$S_{(i+1)2}$，$S_{(i+1)3}$，…$S_{(i+1)n}$で表わす。また、節点S_{ip}と$S_{(i+1)q}$間の距離をｄ（S_{ip}，$S_{(i+1)q}$）で表わす。このとき、これらの節点間を接続して、三角パッチを生成するアルゴリズムのフローを図２．10に、実行例を図２．11に示す。本アルゴリズムは、簡単なヒューリスティックであり、ｄ（S_{ip}，$S_{(i+1)(q+1)}$）とｄ（$S_{i(p+1)}$，$S_{(i+1)q}$）を比較して、距離の短い節点相互間を接続し、できるだ

け正三角形に近い三角パッチを生成しようとするものである。

本再構成方法による立体形状のデータは、各節点の3次元座標値、及び、各三角パッチを構成する3節点のリストである。

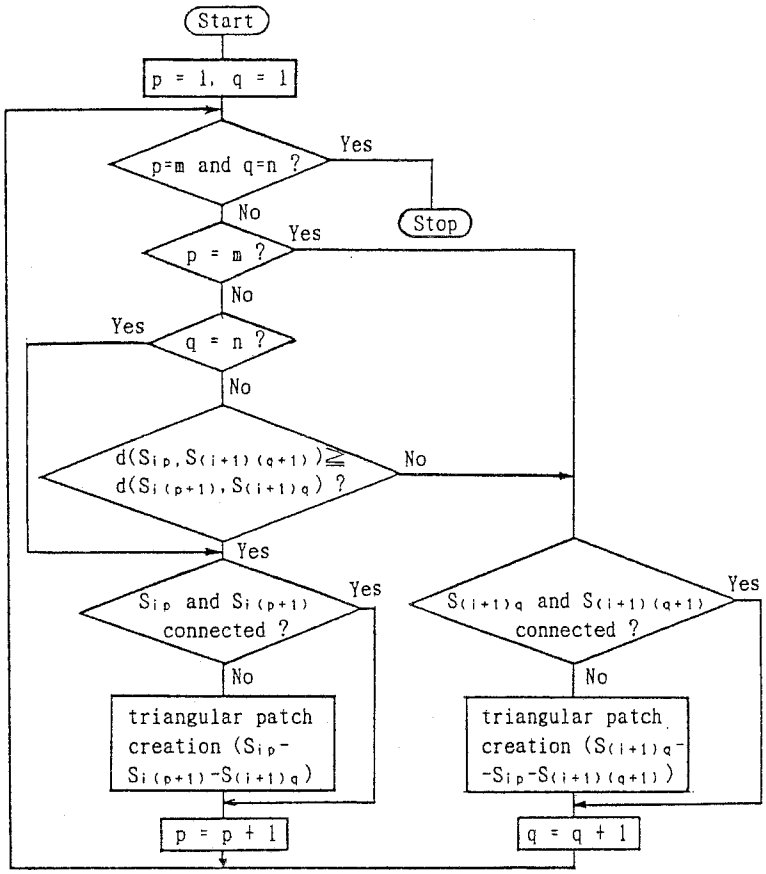


図2.10　三角パッチの生成アルゴリズム

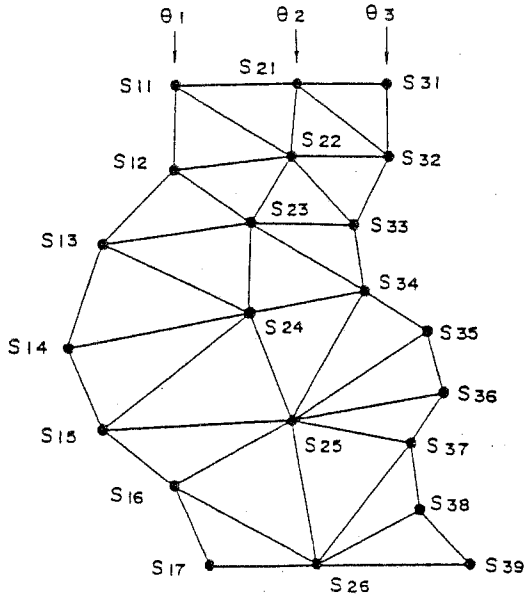
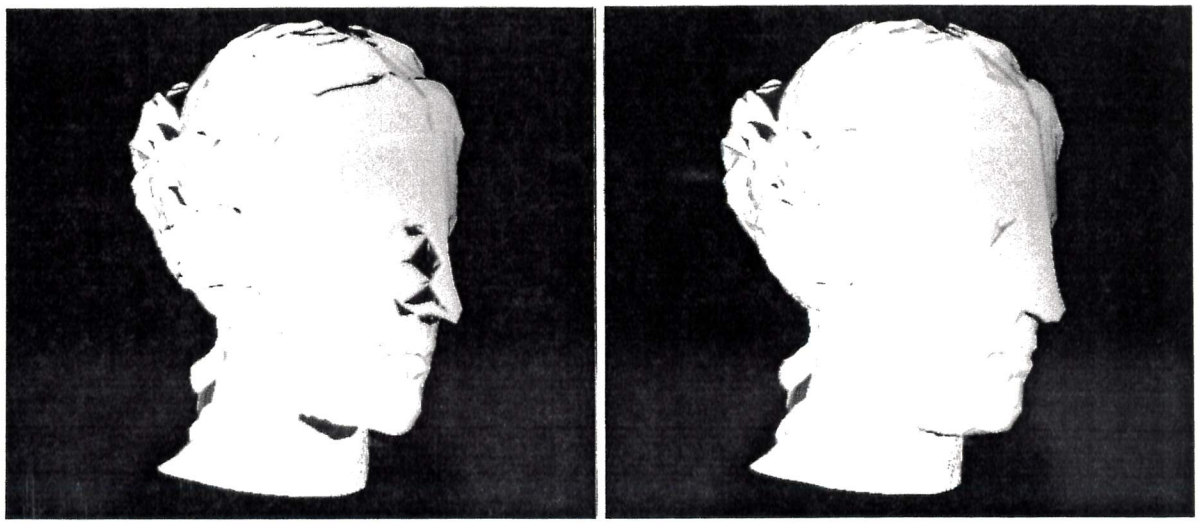


図2.11　三角パッチの生成例

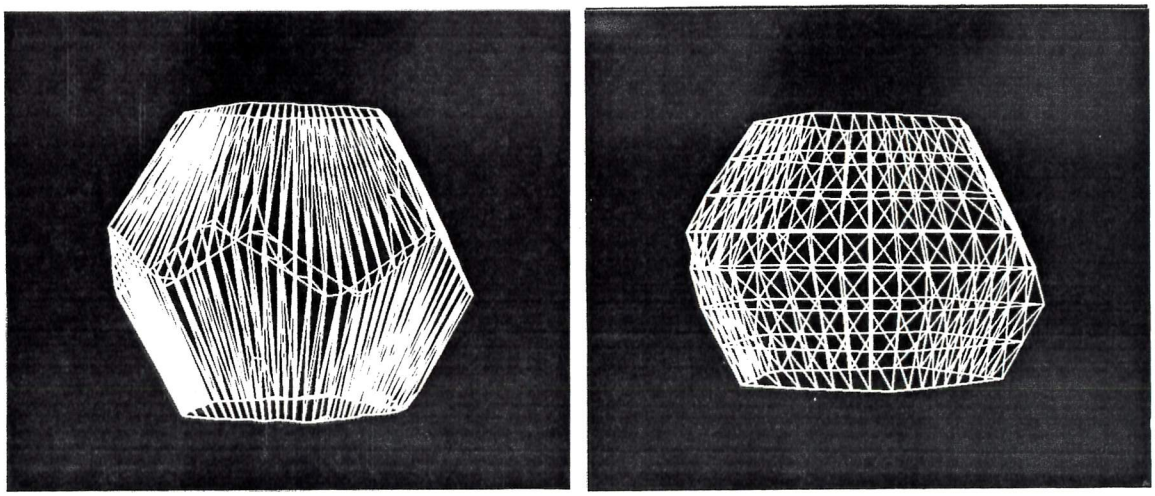
2．3　実験と検討

図2.12(a)は、図2.3のビーナス像を1台のカメラを使って計測した結果をもとに再構成した立体モデルであり、図2.12(b)は、同じく2台のカメラを使って計測・再構成したモデルである（$\Delta\theta=3$[deg]）。1台のカメラでは、鼻等で死角になり計測できなかった部分が存在するが、2台のカメラを用いることにより著しく改善されることがわかる。なお、図2.12(a)の立体モデルは、節点数が2716、三角パッチ数が4656であり、図2.12(b)は、節点数が2831、三角パッチ数が4292である。



(a)　　(b)

図2.12　ビーナス像の入力結果



(a)　　(b)

図2.13　正12面体の入力結果

また、折れ線近似による節点発生の効果を確認するため、正12面体の立体形状を本計測システムで計測し、三角パッチによる立体モデルで再構成した。図2.13(a)は、本折れ線近似アルゴリズムで発生させた節点をもとに再構成した立体モデルのワイヤフレームであり、図2.13(b)は、変形スリット像をv方向に一定の間隔でサンプリングした点から節点を発生させた立体モデルのワイヤフレームである。双方の形状データは、ほぼ同じデータ量（図2.13(a)では、節点数256・三角パッチ数368、図2.13(b)では、節点数288・三角パッチ数504）をもっている。図2.13(a)の形状では、各稜線が正確に再現されているが、図2.13(b)では不明瞭になっているのがわかる。

このように回転テーブルの各回転角で計測したデータを折れ線で近似し、各折れ線の端点から節点を発生させた結果、十分少ないデータ量で立体形状を表現することができ、また、立体の稜線、頂点等の特徴点も保存された。ただし、ここで述べた再構成方法は、計測した回転テーブルの各回転角度ごとのデータ圧縮であり、回転方向については、均等角度（$\Delta\theta$）のサンプリングのままである。したがって、半径rの値が小さな部分ほど三角パッチが密に生成されてしまう問題点が残されている。この回転方向も含めた全体的な立体モデルの再構成方法の検討は、今後の課題である。

3．形状欠落部分の補完方法

3．1　形状欠落部分の補完

第3章では、計測できなかった欠落部分の形状データを補って完全な立体形状を生成する方法を検討する[21][23]。回転テーブルを用いて物体を回転させることは、比較的簡単であるが、物体の任意の部分をスリット光の方向に向けるには、複雑な制御機構が必要であり、また、物体を支える治具の形状データと物体の形状データを分離する必要も生じる。このような理由により、物体の任意の部分をスリット光の方向に高精度で動かして計測することは難しいと判断し、回転テーブル上への物体の置き方を変えて入力した複数の形状データを合成処理して全体形状を生成することにした。具体的には、通常の置き方（以下、メインという）で入力し、そこで欠落した部分を置き方を変えて入力した形状データ（以下、サブという）で補完する。

通常、上部・下部の形状データは、スリット光の当り方が弱いため、計測できないが、たとえ入力できたとしても、筆者らの再構成方法では、半径が小さい部分は、他の部分に比較して高密度に三角パッチが生成される問題点がある。そこで、上部・下部で半径の値が指定した閾値より小さい部分は、入力できてもそのデータを捨て、サブのデータから形状を再構成できるようにした。

メイン形状の上部・下部の欠落部分を補完するときは、約90[deg]傾けた状態でサブ形状を入力し、他の欠落部分を補完するときは、その部分が最もよく計測されるような置き方をする。このようにメイン・サブの形状データは、対象物体を回転テーブルの上に適当に置いて計測したものであるから、相互の位置関係は何らかの方法で求める必要がある。

本形状補完処理アルゴリズムを実行する前提条件として、次のものがある。

(1) メイン・サブの形状データは、2．2で述べた方法により、三角パッチの集合で表現されている。

(2) 欠落部分の補完処理には、サブの形状の節点情報のみを使用し、三角パッチ生成は、本アルゴリズムの中で行う。

(3) サブ形状の節点は、2．2．1で述べた方法により発生させられ、回転テーブルの各回転角度θ_iで計測したデータ上にのみ存在する。

この(3)の前提条件があるため、本アルゴリズムは、任意の三角パッチの集合で表現された形状どうしの合成処理には適用できない。しかしながら、筆者らによる形状計測法と再構成法の特徴をうまく利用して、少ない計算コストで形状補完処理を行う。

本形状補完アルゴリズムの処理の全体的な流れを図3．1に示す。以下、図3．

1の各処理について説明する。

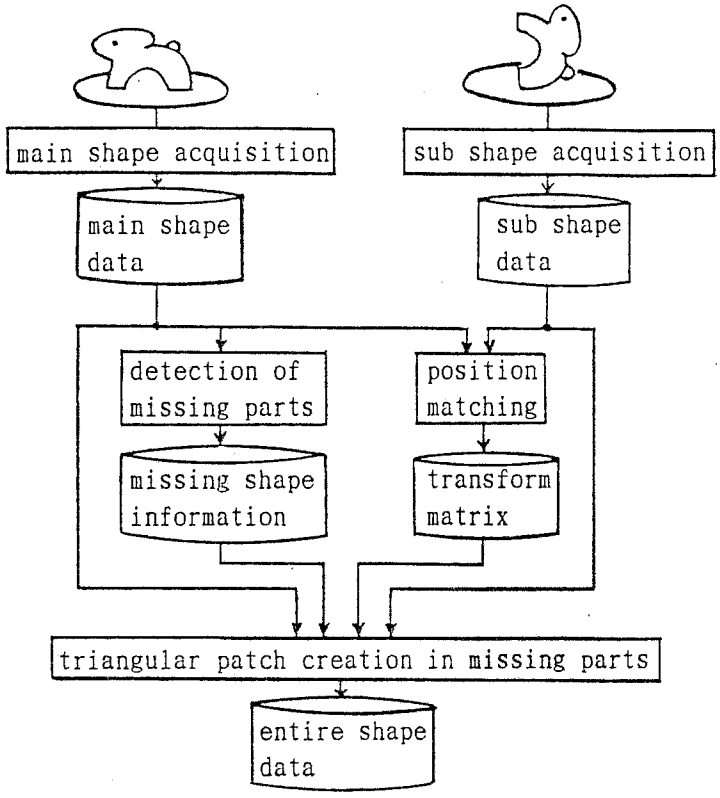


図３．１　形状補完処理のフロー

３．２　欠落部分の補完アルゴリズム

３．２．１　欠落部分の検出

メインの形状データは、三角パッチの集合で表現されている。形状データが欠落していない部分の辺は、２個の三角パッチに共有されるが、データ欠落部分との境界の辺は、１個の三角パッチにしか含まれない。この性質を使って、すべての辺の集合から１個の三角パッチにしか含まれていない辺を抽出し、その辺の系列（３次元上の多角形）で欠落部分を検出する。

３．２．２　位置合わせ

メイン・サブの形状データは、ともに欠落部分のある不完全なものであるが、物体形状のほとんどの部分は、双方の形状データに含まれる。この双方の形状データにともに含まれる部分の形状情報を使って、マウスを使った人手の入力により、メインとサブの形状データの位置関係を求める。

グラフィックディスプレイ上にメイン・サブの２形状データをワイヤフレームで表示し、マウスによる入力で、サブのワイヤフレームを移動させて、メインのワイヤフレームと位置を合わせる。本ソフトウェアは、互いに直角なｘ・ｙ・ｚ

軸の各方向からの平行投影による画像を表示することができ、表示されている2次元の平面内で、サブの形状を平行・回転移動させることができる。位置合わせの要領は、x・y・zの3方向からの各画像で、メイン・サブの各ワイヤフレームの輪郭線形状がともに一致するように、サブの形状を移動させるものである（図3.2参照）。

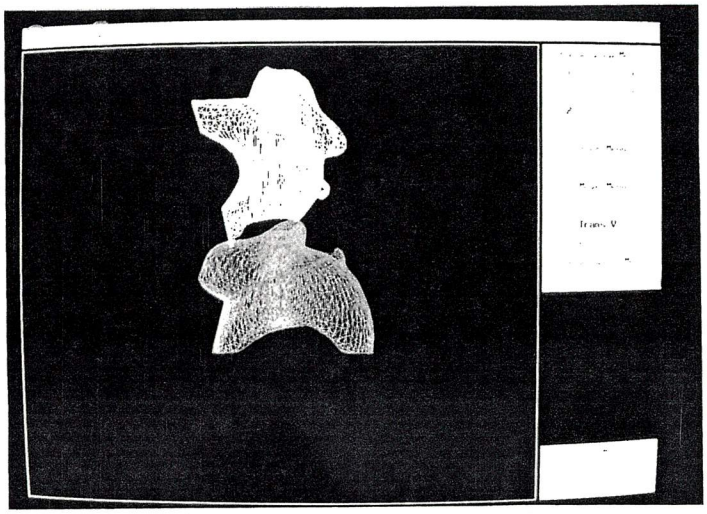


図3.2　位置合わせ画面例

以上の操作により、サブ形状の節点の座標をメイン形状の表面上の対応する点に同次座標変換するマトリクスの各要素、及び、逆にメイン形状の節点データをサブ形状の表面上の対応する点に同次座標変換するマトリクスの各要素が求められ、位置合わせを行うことができる。

3.2.3　欠落部分内部の三角パッチ生成

サブ形状を構成する節点のうち、メイン形状の欠落部分内部にあるものを抽出してメイン形状の節点として加え、さらに、節点間の接続により欠落部分に三角パッチを生成する[3][6]。この処理をサブ形状の座標系における（θ, z）の仮想平面上で2次元的に行った（図3.3参照）。

Step1：メイン形状のデータ欠落部分（多角形）の節点座標を、位置合わせ処理で求めた座標変換マトリクスを使って、サブ形状と同一の座標系（円柱座標系（r, θ, z））に変換する。図3.3(a)の多角形が座標変換された欠落部分を表わしている。

Step2：サブ形状の節点の中で、欠落部分の内部にある節点を求める。その方法は、欠落部分の多角形をトレースして、$\theta = \theta_i$の直線との交点を求め、その座標値から欠落部分の内部にある節点だけを抽出する。

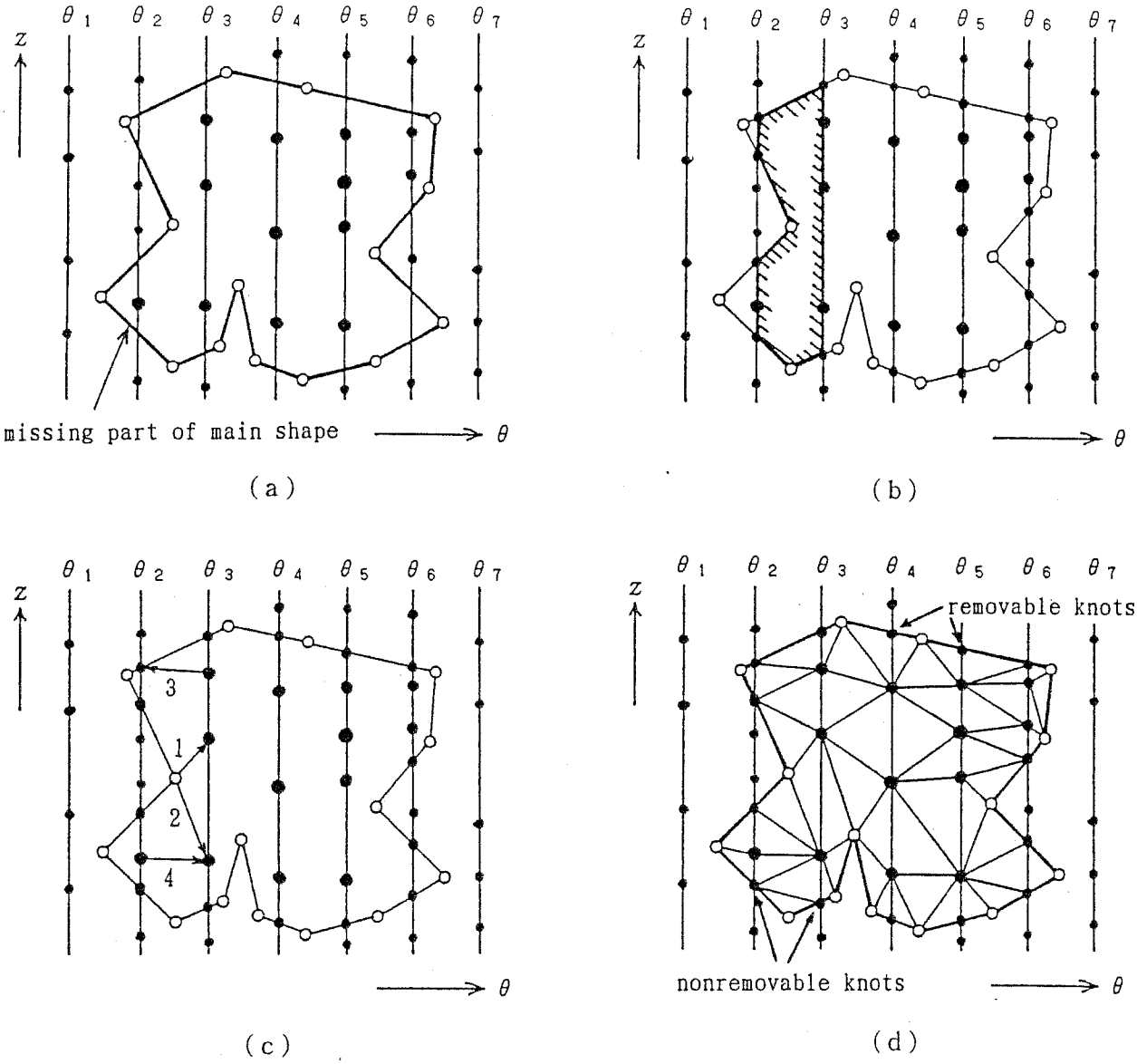


図３．３　欠落部分内部の三角パッチ生成

Step３：$\theta=\theta_i$の直線との交点も一時的に節点とし、データ欠落部分の多角形をθ_iとθ_{i+1}の間隔ごとの複数の多角形に分割する。たとえば、図３．３(b)では、θ_2とθ_3の間の多角形を表している。

Step４：Step３で求めたθ_iとθ_{i+1}の間の多角形を三角形分割することにより、欠落部分内部に三角パッチを生成する。その方法は、最も内角の大きい節点と、そこから最も近い節点を接続して、順に２個の多角形に分割するものである。ただし、節点間を接続する辺は、必ず多角形内部にあるものとする。この操作を、すべての多角形が三角形になるまで繰り返す（図３．３(c)参照）。

Step５：Step３で求めた一時的な節点のうち、削除可能な節点を取り除いて、三

角パッチを併合する。削除できない節点に対しては、メイン側の形状データで関係する三角パッチをさらに分割する。ここで、削除可能とは、図３．３(d)に示すように、Step４の処理でその節点から接続された辺がないことである。

Step６：サブ形状の座標系で補完処理が完了したので、各節点の座標をメイン形状の座標系に変換する。

図３．３では、横方向がθ座標、縦方向がｚ座標に対応する。Step４での距離計算をするには、θ方向の距離を与えなければならない。そこで、メイン欠落部分の多角形を構成する節点の半径の平均値を r_m とし、$\Delta\theta \cdot r_m$ を$\Delta\theta$に対応する近似距離として計算した。

また、$\theta_i \sim \theta_{i+1}$ごとの多角形にまず分割し、そのあと、三角形に分割したのは、次の理由による。

(1) 内部に節点を含む多角形を三角形で分割する問題は、多角形を単に三角形で分割する問題に比べ、解法が難しい[6]。

(2) 実際に計測した回転角度θ_i上の計測データが保存される。

３．３　実験と検討

この立体形状の再構成アルゴリズムを実際にインプリメントし、図３．４のウサギの置き物に対して、実験を行った。

図３．５(a)は、図３．４のウサギの置き物の形状を２台のＴＶカメラを使用して計測し（$\Delta\theta = 3$[deg]）、折れ線近似の閾値（D_{th}）を0.6画素として再構成した結果をワイヤフレームで表示したものである。このモデルの節点は1520個、三角パッチは2761個である。このように２台のＴＶカメラを使用することにより、死角となる部分が少なくなり、上部・下部を除いて完全な形状データが入力できている。また、計測データを折れ線近似したため、形状の特徴が保存されている。

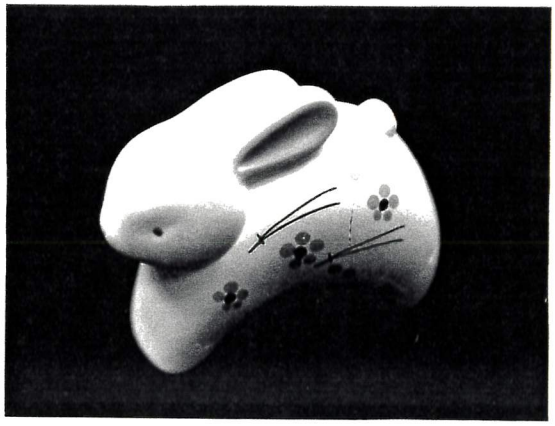
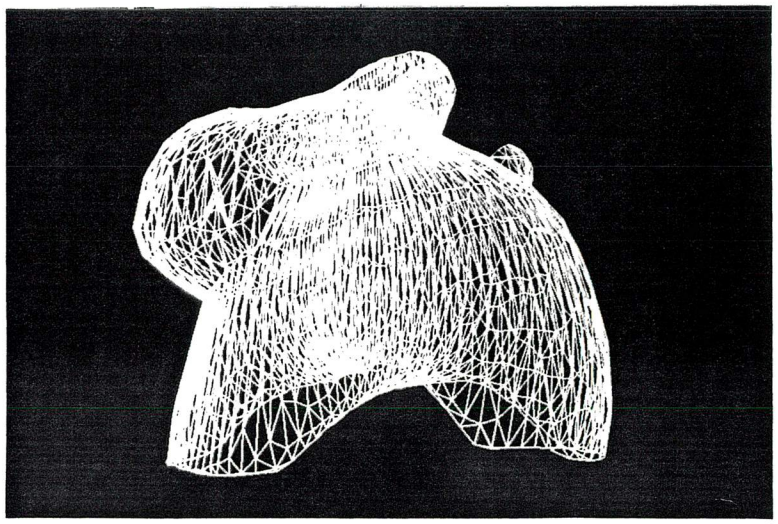
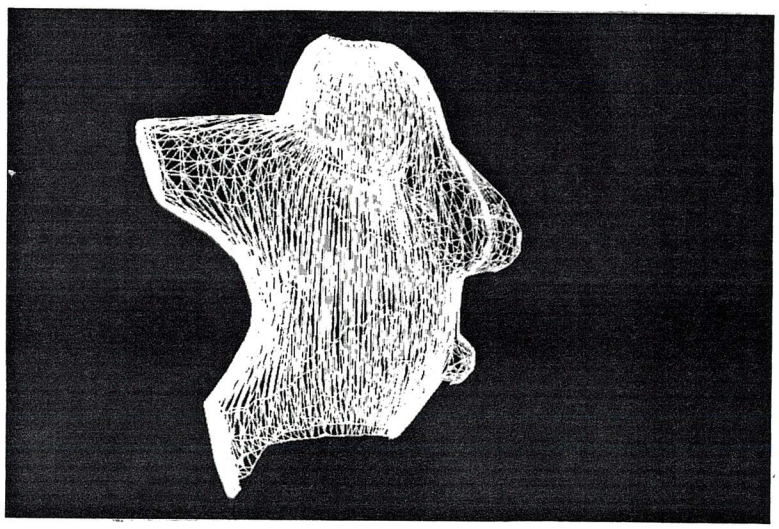


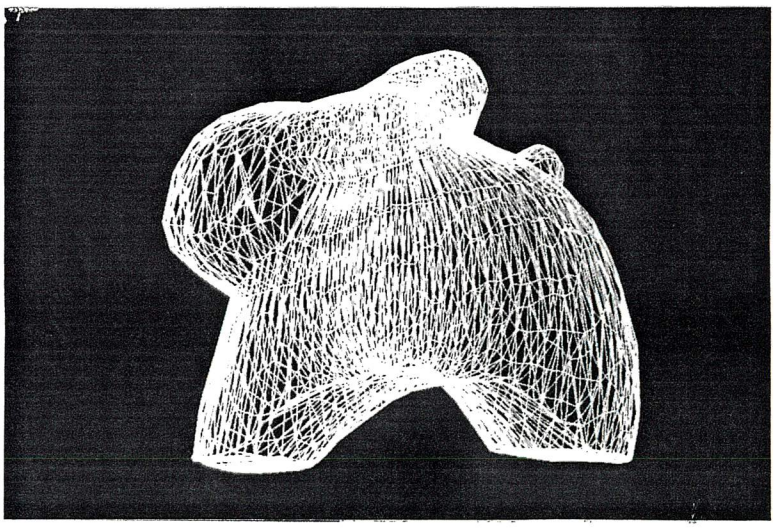
図３．４　ウサギの置き物



(a)



(b)

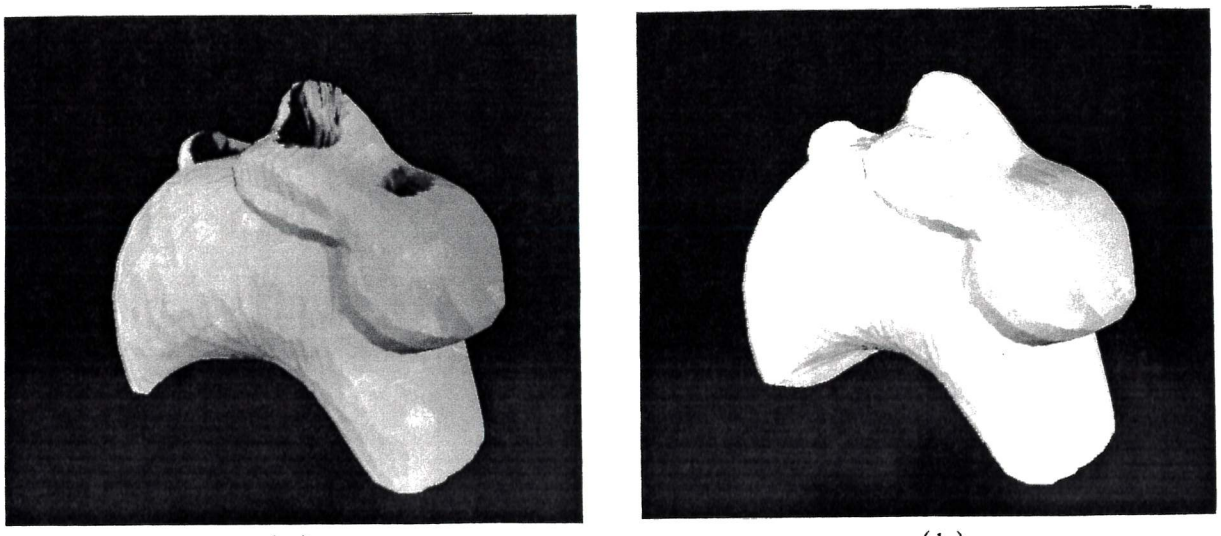


(c)

図３．５　形状補完処理の実験例

次に、形状補完処理により、欠落部分のない完全な形状データ生成の実験を行った。図３．５(a)の形状データ（メインとする）に対して、３．２．１で述べた方法によりデータ欠落部分の検出を行ったところ、上部に３個と下部に１個の欠落部分が検出された。図３．５(b)は、ウサギの置き物を、メイン形状の入力時の姿勢に対して約90[deg]傾けて置いて計測したサブの形状データである。図３．５(c)は、３．２．２の方法で位置合わせをした後、メイン形状の４個の欠落部分をサブ形状データで補完処理した結果である。メインとサブの形状データに計測誤差があると、位置合わせがうまくいかず、また、貼り合わせ部分に段差が生じるが、目視による限り、段差は確認されなかった。なお、この形状データの節点は1851個、三角パッチは3698個である。

また、図３．６(a)は、メイン形状に対するシェーディング処理の結果であり、図３．６(b)は、同じく形状補完処理後の結果である。本形状補完処理により、メイン形状の欠落部分が完全に補完され、ウサギの置き物の全体形状が忠実に再構成されているのがわかる。



(a)　　(b)

図３．６　シェーディング

形状補完処理方法の大きな特徴は、貼り合わせ処理をサブ形状の座標系における（θ，ｚ）の仮想平面上で行ったことである。厳密には、３次元空間上で三角パッチを生成するのがよいが、計算が複雑になり、かつ、不都合な三角パッチが生成されているかどうか（正しく物体表面を再構成しているかどうか）のチェック等、困難な問題点が多い。そこで、２次元的な処理で近似する必要が生じるが、筆者らの計測方法は、（θ，ｚ）に対して半径ｒを計測するものであるから、サブ形状の（θ，ｚ）の平面を利用すれば、サブ形状のすべての計測データを表わ

すことができて都合がよい。このほか、次のような利点がある。

(1) メイン形状で欠落した部分のデータはサブ形状ではうまく入力できているという前提に立てば、欠落部分の多角形の座標をサブ座標系の（θ, z）平面上に変換すると大きな多角形になり、精度よく補完処理ができる。

(2) サブの各回転角θ_iでの節点が$\theta=\theta_i$の直線上に並ぶため、欠落部分の外周とこれらの直線との交点座標が簡単に求められ、その結果、サブ形状の節点が、欠落部分内部にあるかどうかの判定が容易になる。

このように2次元的に処理しても、節点の座標値に関しては、誤差はまったく発生しない。ただ、三角パッチの生成に近似処理が入るが、サブ形状データの補完処理に使用された部分の半径rの分布が、広範囲にわたっていない限り、特異な形状の三角パッチは生成されない。本形状合成アルゴリズムが有効に機能するのは、メイン形状で欠落した部分のデータが、サブ形状の入力においてうまく計測できる場合である。（θ, z）に対して半径rの値が一意に決定しないような物体でも、その欠落部分のデータがサブ形状においてうまく入力できれば、全体形状の再構成が可能である。

3．4　位置合わせ自動化の検討

3．2で述べた欠落部分の補完アルゴリズムでは、メイン・サブの2形状データの位置合わせ部分（3．2．2）がマウスによる人手の入力で行っており、自動化されていない。この位置合わせ処理の自動化について検討した。

3．4．1　位置合わせ自動化の方法

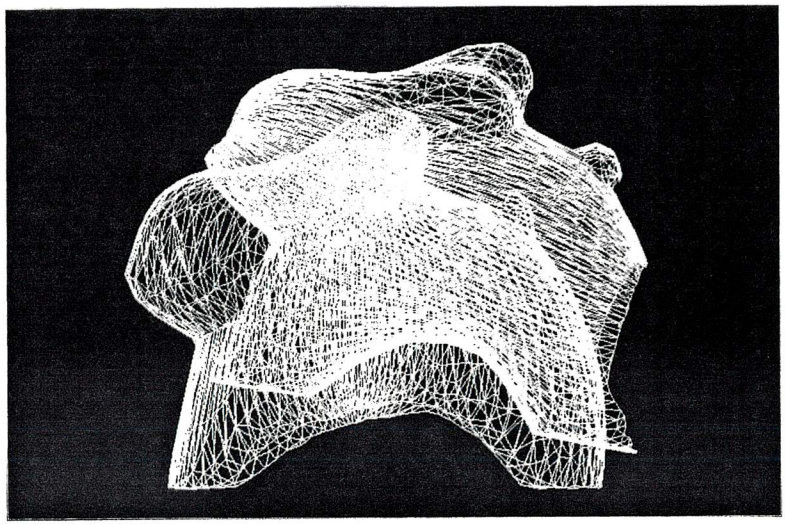
メイン・サブの2形状データの位置を合わせるには、x・y・z各方向の移動、及び、x・y・z軸方向の回転の合計6個の変数の値を求める必要がある。ここでは、この位置合わせ処理を「大まかな位置合わせ」と「正確な位置合わせ」の2段階に分けて処理することとし、正確な位置合わせを中心に検討した。正確な位置合わせの方法は、最適化の一手法である山登り法[14][17]を用いる。山登り法では、ある評価関数を決め、その評価関数が改善される方向に変数の値を少しずつ変化させていき、もはやどの変数の値を変化させても評価関数の値が改善されないとき、そのときの解を近似最適解とする。山登り法を適用する問題の性質にもよるが、一般に初期解が真の最適解にある程度近ければ、山登り法により真の最適解に収束する。ここで提案する方法では、まず、山登り法で必ず真の最適解に収束する範囲に収まるまで、「大まかな位置合わせ」で位置を合わせておき、そのあと、山登り法による「正確な位置合わせ」を実行する。

山登り法の評価関数は、3．2．2において人手で行った方法と同様に、x・y

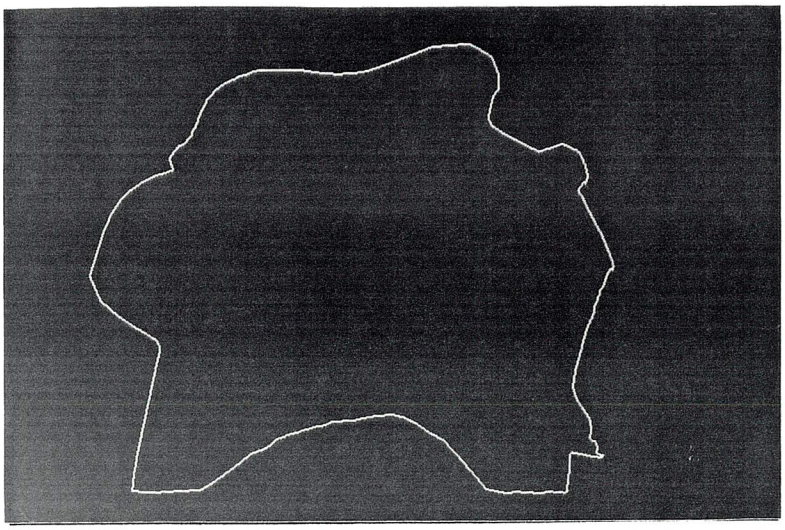
・z軸方向からの平行投影による各画像でメインとサブのワイヤフレーム像の面積和を求め（S_x、S_y、S_z）、その面積和の積とした。

$$f = S_x \times S_y \times S_z \qquad (3.1)$$

この評価関数fの値を最小にするように、サブの形状を移動・回転させる。図3.7(a)は、メインとサブの形状のワイヤフレームを平行投影により表示した例である。図3.7(b)は、この2形状のワイヤフレームの輪郭線である。このような輪郭線内部の面積をx・y・z軸方向からの各画像で求め、評価関数fはその面積の積で表わされる。x・y・z軸方向からの各画像で、メインとサブの形状が完全に一致したとき、評価関数fは、最小値をとる。



(a)



(b)

図3.7　位置合わせ自動化の画面例

メイン・サブの形状はともに欠落部分をもつ不完全なものであるから、位置合わせが完全にできたとき、評価関数ｆの値が最小になる理論的な保証はない。しかし、通常、物体のほとんどの部分はメイン・サブの双方の形状データに含まれており、(3.1)式の評価関数ｆを用いることにより、ほとんどの場合、完全な位置合わせが可能であるといえる。

山登り法による処理では、ある解の近傍で評価関数ｆの値を改善するような解を探索する。このとき、評価関数ｆの値がもっとも改善される方向（勾配ベクトル）を見つけ、その方向に複数の変数を同時に変化させる方法（最急降下法）も考えられるが、今回は、一度には１個の変数しか変化させない方法をまず検討した。変数としては、移動と回転の合計６個あり、それぞれ＋－の方向があるので合計12方向となる。このうち、回転の中心は、その立体モデルを構成する全節点の重心をその立体モデルの重心であると近似し、その座標を中心に回転させることにした。一回の移動量・回転量（歩み幅）は固定とし、その歩み幅で収束すれば、より小さな歩み幅でさらに収束させる方法をとった。すなわち、複数ステップの歩み幅を設定し、最終の歩み幅で収束したときに近似最適解とした。

この山登り法で位置合わせを行うには、大まかな位置合わせが前提である。この方法としては、具体的には検討できなかったが、次の方法が考えられる。

(1) まず、メイン・サブの各々について全節点の重心を計算して求めた近似重心を一致させる。そのあとは、その重心を中心として、回転させることになるので、残る変数は３個となる。この方法の問題点として次の項目がある。双方の形状が不完全であり、かつ、全節点の重心を物体の重心に近似したため、近似重心にどの程度の誤差があるか。また、回転方向をどのようにして一致させるか。

(2) 物体表面上の３点を認識できれば、その３点を一致させることにより位置合わせを行うことことができる。３特徴点を自動的に抽出できれば問題ないが、現実には無理である。そこで、対象物体に３個のマークを貼り、ステレオ視によって、そのマーク座標を求める方法等が考えられる。

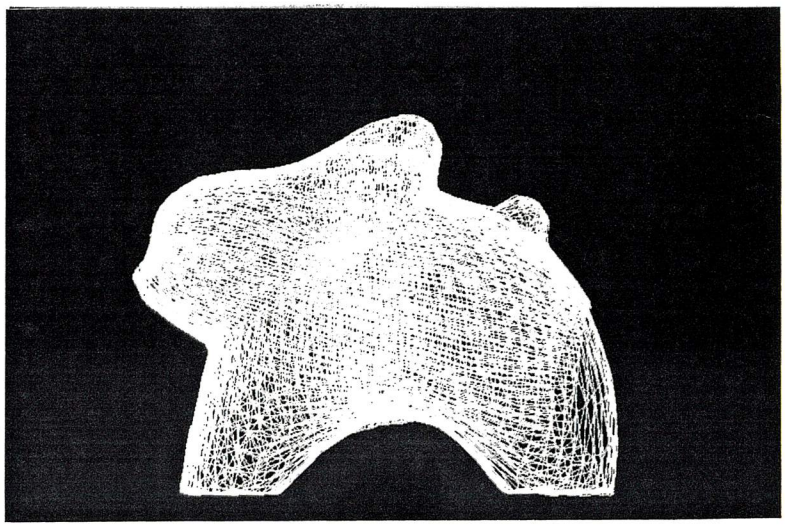
(3) 物体の最長の軸が、双方の立体モデルでともに入力できていれば、その軸を合わせることにより、その軸回りの回転方向の変数のみが残る。任意の物体に対して最長の軸が一意に決定できるとは限らず、この方法はすべての場合に適用できるとは限らない。

３．４．２　位置合わせ自動化の実験

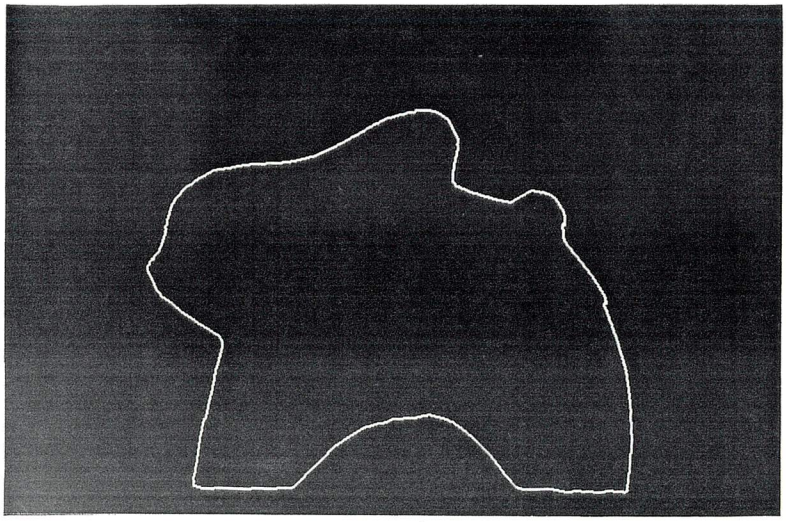
この山登り法による位置合わせアルゴリズムをインプリメントし、初期解がどの範囲であれば最適解に収束するかを中心に検討した。ここでは、図３．４のウサ

ギの置き物（全長約10cm）を入力した図３．５(a)（メイン）と図３．５(b)（サブ）の形状データを用いて実験を行った。

メイン・サブ形状のワイヤフレームによる投影像を512×512のサイズのディジタル画像上に描き、その外側の輪郭線内の画素数を数えることで面積を算出した。図３．８(a)は、最適位置に収束したときの正面からの画像であり、図３．８(b)は、そのときの輪郭線である。若干の計測誤差があるため、完全には一致しない。



(a)



(b)

図３．８　位置合わせ収束時の画面例

［実験１］

最適位置からサブの形状を、ｘ・ｙ・ｚの各方向に５～15[mm]移動させ、さらに重心を中心として各方向に５～15[deg]回転させた位置を初期解として、山登り

法を適用し、最適解に収束するかどうかを実験した。歩み幅は、移動量が3.0・1.0・0.3・0.1[mm]、回転量が3.0・1.0・0.3・0.1[deg]の4ステップとした。ランダムに初期解を6個発生させて実験したが、いずれも最適解に収束した。

［実験2］

実験1と同様に、サブの形状をx・y・zの各方向に15～25[mm]移動させ、さらに重心を中心として各方向に15～25[deg]回転させた位置を初期解とした。歩み幅は実験1と同じである。ランダムに初期解を6個発生させて実験した。この結果、5例は最適解に収束したが、1例は最適解でない解に収束した。

物体の形状により最適解への収束の傾向が異なると思われるが、以上の実験結果から、ここで取り扱ったウサギの置き物に関する限り、距離で全長の10%程度、角度で10[deg]程度以内の誤差で大まかな位置合わせを予め行っておけば、山登り法により最適な位置合わせに収束することが確かめられた。

最適な位置合わせができているときのメイン・サブの近似重心（全節点の重心）のずれを計測した結果、近似重心のずれは、6.9[mm]であった。この値は、全長の10%以内には収まっているが、かなり大きな値である。また、各形状の近似重心から最も遠い節点を求めたところ、メインとサブで一致しなかった。このような状況から大まかな位置合わせの方法は、かなり難しいと判断される。

今後の課題として、大まかな位置合わせの方法の検討、山登り法の収束加速方法の検討等が挙げられる。

4．物体表面の平面の抽出方法

4．1　Ｈｏｕｇｈ変換による特徴抽出

画像処理やコンピュータビジョンにおいて、２次元上にランダムに分布した点の集合から直線・円等の特徴を抽出するのにＨｏｕｇｈ変換がよく用いられる。直線・円・楕円等のパターンを抽出するＨｏｕｇｈ変換や、任意形状の（式では表現できない）特徴を抽出する一般化Ｈｏｕｇｈ変換（generalized Hough transform）においても、その特徴を精度よく抽出するには、膨大な計算時間と計算メモリが必要である。最近では、計算量をいかにして低減し、Ｈｏｕｇｈ変換を実用的なものにするかが研究対象となっている[12][26]。一方、Ｈｏｕｇｈ変換を３次元に拡張すると、３次元空間上に分布した点の集合から平面・球・円錐等の特徴を抽出する手段となる[15][24]。２次元のＨｏｕｇｈ変換に対して、このような次元数を拡張したＨｏｕｇｈ変換は、多次元化Ｈｏｕｇｈ変換（extended Hough transform）と呼ばれている[12]。３次元Ｈｏｕｇｈ変換は、さらに多量の計算量を必要とする問題点はあるが、発見的な手法と比較すれば、局所的な特徴を拡張するのではなく、３次元空間全体を見渡して、特徴を抽出するという長所がある。３次元Ｈｏｕｇｈ変換で平面を抽出しようとするアプローチは、これまでに中沢ら[15][24]の例があるだけであり、計算量の問題があるため、あまり行われていない[12]。

ここでは、光切断法による本形状計測システムで得られる３次元データをもとに、３次元Ｈｏｕｇｈ変換の手法を使って、物体表面上の平面を抽出することを検討する。光切断法では、変形スリット像上の直線部分を検出し、その直線部分を３次元空間上に変換すると、物体表面上の直線データを正確に検出できる。この事実に着目し、計測した物体表面の直線データの集合から平面を抽出することを検討する[22]。また、本計測システムで得られる直線データの規則性を利用したさらに効率的な手法を提案する。

4．2　３次元Ｈｏｕｇｈ変換による平面抽出方法

まず、４．２．１では、従来からの方法による３次元空間上の点の集合からの平面抽出法について簡単に説明する。そのあと、３次元空間上の直線を直接Ｈｏｕｇｈ空間上に変換する３次元Ｈｏｕｇｈ変換（以下、直線型３次元Ｈｏｕｇｈ変換と呼ぶ）を用いた平面抽出法について、４．２．２では、任意の直線集合から平面を抽出する場合、４．２．３では、本形状計測システムで得られる直線集合から平面を抽出する場合を述べる。

4．2．1　点の集合からの平面抽出法

3次元空間（x，y，z）上の平面は、図4．1に示すように、原点から平面へ下ろした垂線と平面の交点を極座標表現で（θ，ϕ，ρ）とした場合、(4.1)式で表される。

$$\cos\theta\sin\phi\ x + \sin\theta\sin\phi\ y + \cos\phi\ z = \rho \qquad (4.1)$$

ここで、3次元空間（x，y，z）と3次元Ｈｏｕｇｈ空間（パラメータ空間）（θ，ϕ，ρ）の対応を(4.1)式上で考える。3次元Ｈｏｕｇｈ空間（θ，ϕ，ρ）上の1点を指定すれば、3次元空間（x，y，z）上の1平面が対応する。また、図4．2に示すように、3次元空間（x，y，z）上の1点を指定すれば、3次元Ｈｏｕｇｈ空間（θ，ϕ，ρ）上では、(4.2)式の曲面が対応する。この曲面は、（x，y，z）空間上の指定した1点を必ず含むような平面の集合であると考えられる。したがって、（x，y，z）空間上の複数の点が、もし、同一平面上に存在すれば、それらの各点に対応する（θ，ϕ，ρ）空間上の曲面は、1点で交わる。また、この（θ，ϕ，ρ）空間上の交点（$\hat{\theta}$，$\hat{\phi}$，$\hat{\rho}$）から、もとの平面の方程式を求めることができる。

この性質を使って、3次元空間（x，y，z）上に点p_i（x_i，y_i，z_i）の集合Ｐがあり、かつ、集合Ｐの部分集合Ｐ'の点が同一平面上に存在する場合、次の手順により、平面の方程式を得ることができる。

Step1：集合Ｐの各点p_i（x_i，y_i，z_i）を(4.2)式の3次元Ｈｏｕｇｈ変換の変換式で3次元Ｈｏｕｇｈ空間（θ，ϕ，ρ）上の曲面に変換する。

$$x_i\cos\theta\sin\phi + y_i\sin\theta\sin\phi + z_i\cos\phi = \rho \qquad (4.2)$$

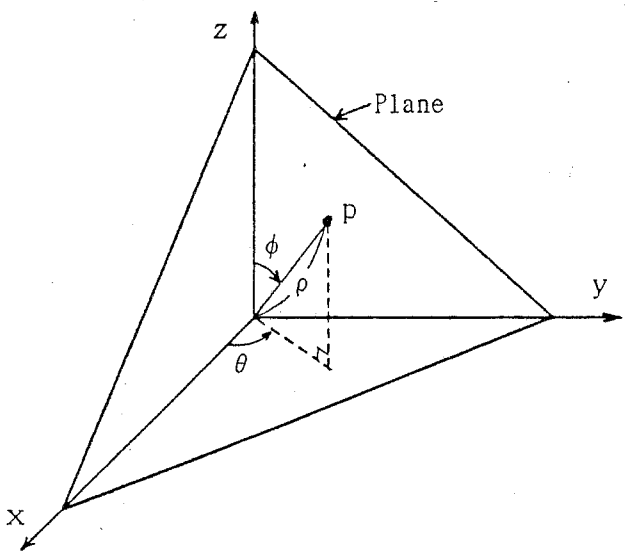


図4．1　3次元空間（x，y，z）上の平面

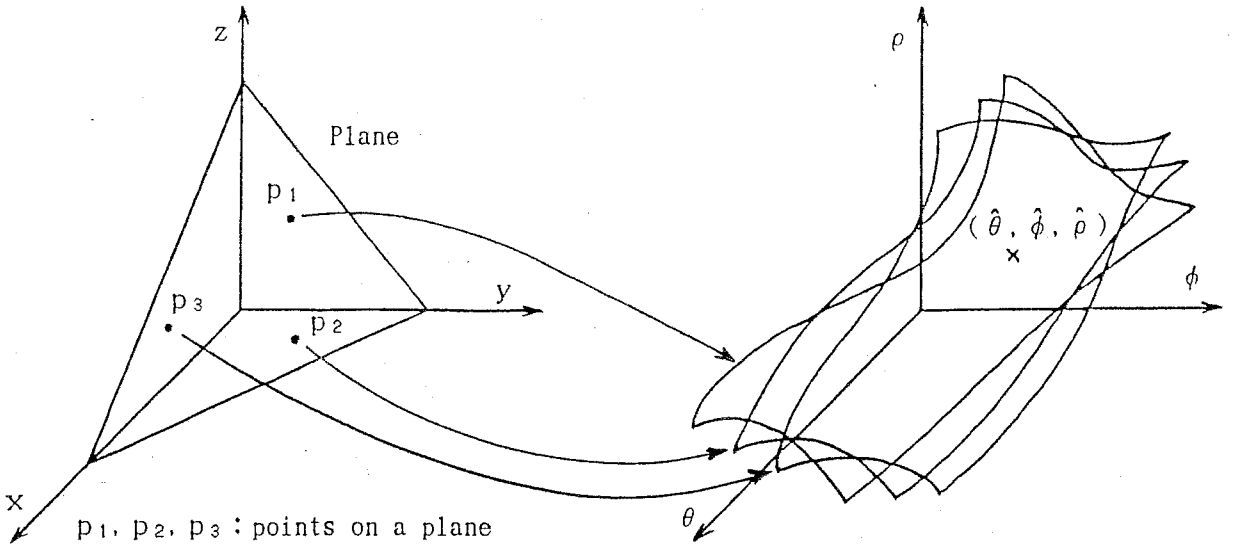


図４．２　３次元Ｈｏｕｇｈ変換による平面抽出の原理

Step２：３次元Ｈｏｕｇｈ空間上で、多数の曲面が交わっているピーク点（$\hat{\theta}$, $\hat{\phi}$, $\hat{\rho}$）を検出し、(4.3)式を使って平面の方程式を算出する。

$$\cos\hat{\theta}\sin\hat{\phi}\,x + \sin\hat{\theta}\sin\hat{\phi}\,y + \cos\hat{\phi}\,z = \rho \qquad (4.3)$$

ここで、３次元Ｈｏｕｇｈ空間（θ, ϕ, ρ）上で曲面が交わるピーク点を探索する範囲は、(4.1)式の平面方程式の表現方法と三角関数の周期性より、(4.4)式の範囲である。

$$\begin{aligned} &0 \leqq \theta < 2\pi \\ &0 \leqq \phi \leqq \pi/2 \\ &-\infty < \rho < \infty \end{aligned} \qquad (4.4)$$

本手順は、（ｘ，ｙ，ｚ）空間上の１点１点について、（θ, ϕ, ρ）空間上での対応する曲面を求め、そのあと、曲面間のピーク点を検出する必要がある。これをディジタル的に実現するには、３次元Ｈｏｕｇｈ空間（θ, ϕ, ρ）を細かい画素（voxel）の集合に分割して、曲面をその上に展開する。そして、各画素ごとに、何個の曲面がその画素を通ったかを数え上げる。したがって、３次元Ｈｏｕｇｈ空間の分割幅が、そのまま平面の検出分解能に対応し、精度よく平面を抽出するには、分割幅を小さくする必要があり、多大な計算時間とメモリーを要する問題点がある。

4．2．2　任意の直線集合からの平面抽出法

3次元空間上の任意の直線集合が与えられたとき、その直線の部分集合で構成される平面の抽出方法について検討する。3次元空間（x，y，z）上の点p_iは、Hough空間（θ，ϕ，ρ）上では曲面に対応した。ここでは、まず、（x，y，z）空間上の直線は、Hough空間上の曲線に対応することを示す。

［定理 1］

3次元空間（x，y，z）上の直線lは、3次元Hough空間（θ，ϕ，ρ）上では曲線に対応し、この曲線は、直線l上の任意の2点をそれぞれ3次元Hough変換して生成される（θ，ϕ，ρ）空間上の2曲面が交わる交線として求めることができる（図4．3参照）。

（証明）（x，y，z）空間に存在する直線l上の任意の2点をp_1（x_1，y_1，z_1）、p_2（x_2，y_2，z_2）（$p_1 \neq p_2$）とする。この2点p_1、p_2にそれぞれ対応するHough空間（θ，ϕ，ρ）上での曲面は、(4.5)式で表わされる。

$$\begin{aligned} x_1\cos\theta\sin\phi + y_1\sin\theta\sin\phi + z_1\cos\phi &= \rho \\ x_2\cos\theta\sin\phi + y_2\sin\theta\sin\phi + z_2\cos\phi &= \rho \end{aligned} \qquad (4.5)$$

(4.5)式の2曲面は、以下に示すように3次元Hough空間（θ，ϕ，ρ）上の曲線で交わる。

(i) $z_1 \neq z_2$ のとき

$\theta = \theta_0$ に対する（ϕ，ρ）＝（ϕ_0，ρ_0）の値は次の(4.6)式(4.7)式で表わされる。

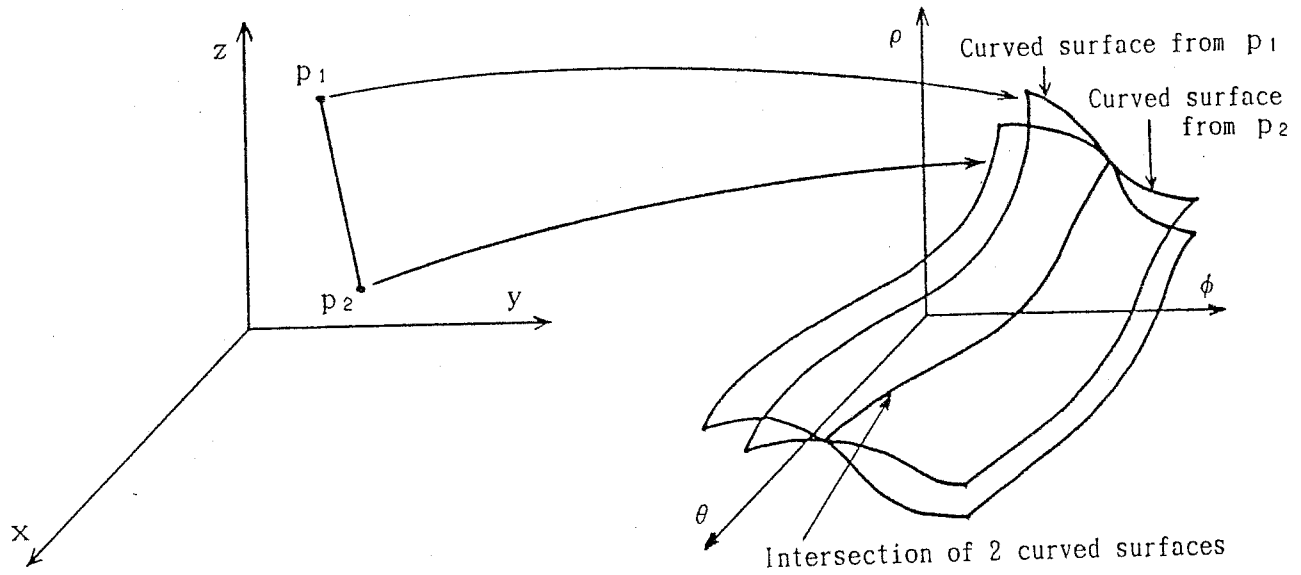


図4．3　3次元空間上の直線のHough変換

$$\phi_0 = \tan^{-1} \frac{z_2 - z_1}{(x_1 - x_2)\cos\theta_0 + (y_1 - y_2)\sin\theta_0} \quad (4.6)$$

$$\rho_0 = (x_1\cos\theta_0 + y_1\sin\theta_0)\sin\phi_0 + z_1\cos\phi_0 \quad (4.7)$$

したがって、$\theta = \theta_0$ の値を連続的に変化させると（θ_0, ϕ_0, ρ_0）は、Ｈｏｕｇｈ空間上に曲線を描くことがわかる。

(ii) $z_1 = z_2$ ($= z_0$) のとき

(4.5)式を解けば、(4.8)式、及び、(4.9)式の曲線になる。

$$\phi = 0 \quad \text{かつ} \quad \rho = z_0 \quad (4.8)$$

$$\theta = \tan^{-1} \frac{x_2 - x_1}{y_1 - y_2} \ (= \theta_0 \text{ とする}) \text{ かつ}$$
$$\rho = (x_1\cos\theta_0 + y_1\sin\theta_0)\sin\phi + z_0\cos\phi \quad (4.9)$$

（ｘ，ｙ，ｚ）空間上の２点p_1、p_2を必ず含む平面のパラメータ（θ, ϕ, ρ）は、(4.5)の２式を共に満足する必要があり、その平面パラメータの集合は、このようにＨｏｕｇｈ空間上の２曲面が交わる曲線に対応する。また、２点p_1、p_2を必ず含む平面の集合は、直線ｌを含む平面の集合と明らかに等価である。

以上の結果から、定理１が証明できる。 □

定理１は、次のことからも類推できる。すなわち、３次元空間上のある直線を必ず含むような平面の集合は、この直線を軸として回転させたとき生成される平面の集合で表わされ、３次元空間上の１点を必ず含む場合に比較して、平面の自由度が少なくなる。

定理１の性質を使って、３次元空間（ｘ，ｙ，ｚ）上の直線の集合Ｌから平面を抽出するのに、直線型３次元Ｈｏｕｇｈ変換を使ったアルゴリズム１が成立する。

［アルゴリズム１］

Step１：集合Ｌに含まれる各直線を定理１の方法により、３次元Ｈｏｕｇｈ空間（θ, ϕ, ρ）上の曲線に変換する。

Step２：３次元Ｈｏｕｇｈ空間上で、多数の曲線が交わるピーク点（$\hat{\theta}$, $\hat{\phi}$, $\hat{\rho}$）を検出し、(4.3)式を使って平面の方程式を算出する。 □

点の集合からの平面抽出法では、３次元Ｈｏｕｇｈ空間上に曲面を生成する必要があったが、このアルゴリズム１では、曲線を生成すればよいことになり、この結果大幅な計算量の低減が可能となる。

４．２．３　条件付き直線集合からの平面抽出法

筆者らの形状計測システムでは、スリット光を回転テーブルの回転軸方向に投影しているため、ｚ軸を回転軸とする円柱座標系（ｒ，θ，ｚ）によれば、得られる物体表面上のデータは、特定の偏角θ方向の輪郭線形状である。この形状を折れ線で近似すれば、物体表面上の直線データを正確に検出することが可能である。ここでは、物体を回転させて収集した全周囲のこのような直線データの集合から、物体表面の平面を抽出することを検討する。

この各直線データは、図４．４に示すように、同一の偏角θ上に存在するという性質を持つ。つまり、(4.5)式において点p_1（x_1，y_1，z_1）と点p_2（x_2，y_2，z_2）が、ともに同一偏角θ^*上にあるとすれば、次の(4.10)式の関係が成立する。

$$\begin{aligned} x_1 &= r_1 \cos\theta^* \\ y_1 &= r_1 \sin\theta^* \\ x_2 &= r_2 \cos\theta^* \\ y_2 &= r_2 \sin\theta^* \end{aligned} \qquad (4.10)$$

ここで、$r_1 = (x_1^2+y_1^2)^{1/2}$ 、$r_2 = (x_2^2+y_2^2)^{1/2}$である。

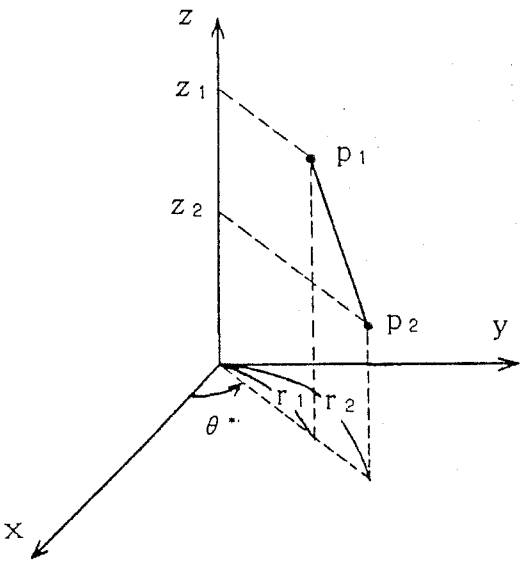


図４．４　本形状計測システムで得られる直線データ

この(4.10)式の条件を持つ直線を直線型３次元Ｈｏｕｇｈ変換する。(4.10)式の関係を(4.5)式に代入すると、(4.11)式が得られる。

$$\begin{aligned} r_1 \sin\phi \cos(\theta - \theta^*) + z_1 \cos\phi &= \rho \\ r_2 \sin\phi \cos(\theta - \theta^*) + z_2 \cos\phi &= \rho \end{aligned} \qquad (4.11)$$

このとき、r_1とr_2の関係、及び、z_1とz_2の関係によって、Ｈｏｕｇｈ空間（θ, ϕ, ρ）上に描かれる曲線は、次のように分類される。

(i) $r_1 \neq r_2$ かつ $z_1 \neq z_2$のとき

$$\rho = \frac{r_2 z_1 - r_1 z_2}{r_2 - r_1} \cos\phi \qquad \text{かつ} \qquad (4.12)$$

$$\tan\phi = \frac{z_1 - z_2}{(r_2 - r_1)\cos(\theta - \theta^*)} \qquad (4.13)$$

この場合のＨｏｕｇｈ空間上に生成されるＨｏｕｇｈ曲線の例を図４．５(a)に示す。このように、Ｈｏｕｇｈ曲線は、(4.12)式で表される曲面と(4.13)式で表わされる曲面が交わる交線となる。

ここで、$(r_2 z_1 - r_1 z_2)/(r_2 - r_1)$の意味を考える。$(r_2 z_1 - r_1 z_2)/(r_2 - r_1) = z_k$ とすると次の(4.14)式が成立する。

$$\frac{z_k - z_1}{r_1} = \frac{z_k - z_2}{r_2} \qquad (4.14)$$

図４．６より、z_kは、直線$p_1 p_2$がｚ軸と交わる交点のｚ座標（ｚ切片）を表しているのは明らかである。

(ii) $r_1 = r_2$ （$= r_0$） かつ $z_1 \neq z_2$のとき

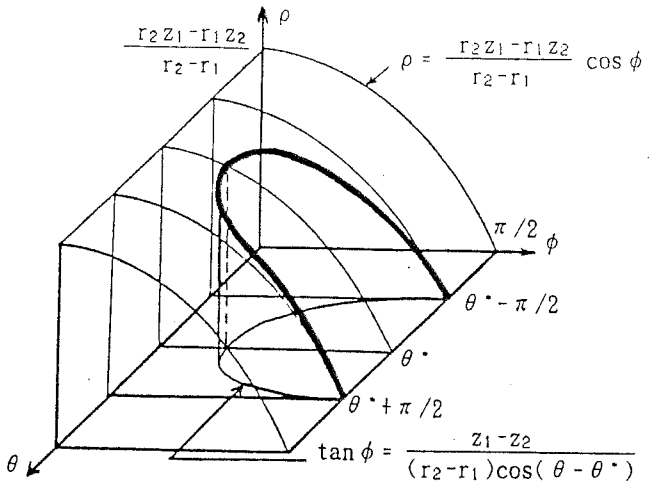
$$\phi = \pi/2 \quad \text{かつ} \quad \rho = r_0 \cos(\theta - \theta^*) \qquad (4.15)$$

この場合のＨｏｕｇｈ曲線の例を図４．５(b)に示す。

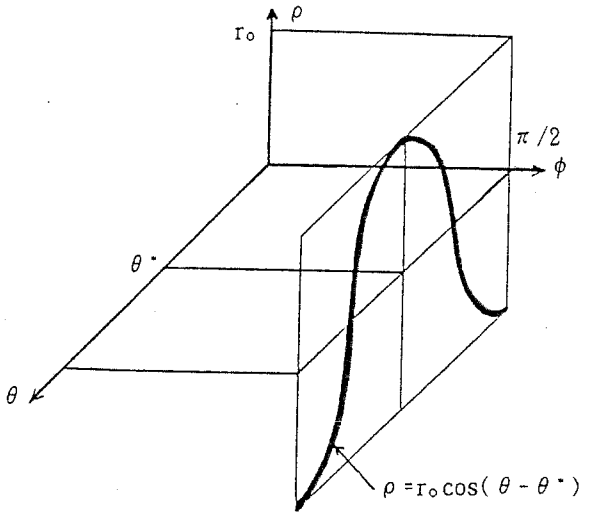
(iii) $r_1 \neq r_2$ かつ $z_1 = z_2$ （$= z_0$）のとき

$$\begin{aligned} &\rho = z_0 \cos\phi \quad \text{かつ} \\ &(\phi = 0 \quad \text{または} \quad \theta = \theta^* + (2n+1)\pi/2) \quad (n : \text{整数})) \end{aligned} \qquad (4.16)$$

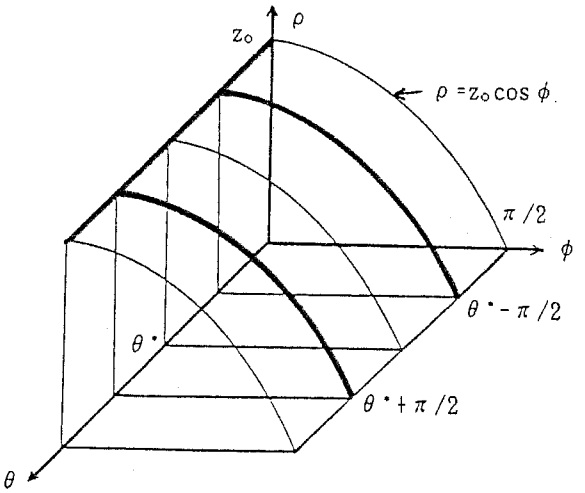
この場合のＨｏｕｇｈ曲線の例を図４．５(c)に示す。



(a) Hough curve when $r_1 \neq r_2$ and $z_1 \neq z_2$
(in case of $\frac{r_2z_1-r_1z_2}{r_2-r_1}>0$ and $\frac{z_1-z_2}{r_2-r_1}>0$)



(b) Hough curve when $r_1=r_2(=r_o)$ and $z_1 \neq z_2$
(in case of $r_o>0$)



(c) Hough curve when $r_1 \neq r_2$ and $z_1=z_2(=z_o)$
(in case of $z_o>0$)

図４．５　Ｈｏｕｇｈ空間上の曲線生成例

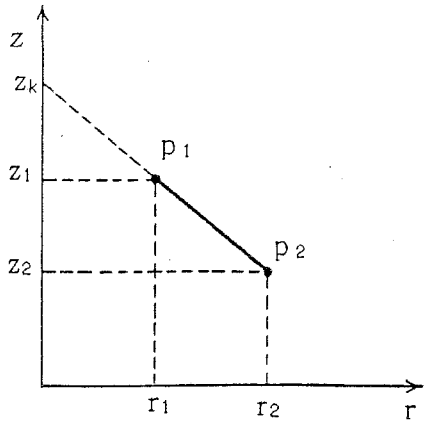


図４．６　直線データのｚ切片

以上のように、(4.10)式の条件を持つ直線をＨｏｕｇｈ空間上に変換すれば、そのＨｏｕｇｈ曲線は、$r_1 = r_2$ となる場合を除いて、(4.12)式の曲面上に生成される。また、$r_1 = r_2$ となる場合も、$\phi = \pi/2$ なる平面上に生成される。直線群が平面を構成するには、直線型３次元Ｈｏｕｇｈ変換により生成されるＨｏｕｇｈ曲線群が１点で交わる必要がある。よって、$\phi = \pi/2$、かつ、$\rho = 0$ となる直線上で交わる場合を除いて、Ｈｏｕｇｈ曲線群が１点で交わるためには、少なくともこれらの曲線群は、同一の(4.12)式の曲面上、あるいは、$\phi = \pi/2$ の平面上になければならない。このことは、各直線のｚ切片の値が等しいことと等価である。また、$\phi = \pi/2$、かつ、$\rho = 0$ となる直線上で交わる場合は、対応する平面がｚ軸を含む場合である。

この性質を使って、各直線は(4.10)式の条件を持つような直線の集合Ｌが与えられたとき、２次元的な手法で平面を抽出する次のアルゴリズム２が成立する（ただし、ｚ軸を含む平面を抽出する場合を除く）。この手法は、直線集合を予めｚ切片の値で分類しておけば、３次元Ｈｏｕｇｈ空間上での曲線間の交点を求める操作が、２次元Ｈｏｕｇｈ空間上での曲線間の交点を求めることに帰着できるという性質を用いるものである。

［アルゴリズム２］

Step１：直線集合Ｌの直線をｚ切片の値（$r_1 \neq r_2$ のときは $(r_2 z_1 - r_1 z_2)/(r_2 - r_1)$）で分類する。

Step２：Step１で分類した直線の各部分集合（同一のｚ切片をもつ）ごとに、各直線を２次元のＨｏｕｇｈ空間（θ，ρ）上の曲線に展開し、多数の曲線が交わるピーク点を検出する。

Step３：検出したピーク点を（$\hat{\theta}$，$\hat{\rho}$）とし、(4.13)式を用いて$\hat{\phi}$の値を計算し（ただし、ｚ切片が ∞ となる直線の部分集合は $\phi = \pi/2$）、（$\hat{\theta}$，$\hat{\phi}$，$\hat{\rho}$）を抽出した平面のパラメータとする。 □

このようにＨｏｕｇｈ曲線間の交点を求めるのに、３次元Ｈｏｕｇｈ空間（θ，ϕ，ρ）上の曲線を（θ，ρ）平面へ投影して行った。この処理は、（θ，ϕ）平面に投影しても可能であったが、（θ，ρ）平面へ投影したのは、次の理由による。

(1) $r_1 = r_2$（$\phi = \pi/2$）のときでも、r_1 と r_2 の値が近い（ϕ が $\pi/2$ に近い）ときでも、精度よく平面を抽出できる。

(2) $\phi = 0$ に近い平面を検出する場合は、精度が悪くなるが、光切断法を使用した筆者らの形状入力方法では、このような平面上の直線データはもともと得られにくい。

(3) 各直線ごとにρの値がとりうる範囲を計算することができる。直線の部分集合ごとに、それに属する直線のρの範囲の和の部分を分割することにより、ρ方向の分割を効率よく設定できる。

４．３　アルゴリズムのインプリメント

直線集合からの平面抽出の有効性、及び、アルゴリズム２の２次元的な処理の有効性を評価するため、アルゴリズム１、及び、アルゴリズム２を実際にインプリメントした。

アルゴリズム１をインプリメントしたときの留意事項は次のとおりである。

(1) ディジタル的に３次元Ｈｏｕｇｈ空間上のピーク点を検出するため、(4.4)式の範囲を図４．７に示すように$\Delta\theta \cdot \Delta\phi \cdot \Delta\rho$の一定の間隔で分割した。ただし、ρ方向は、実用上の絶対値の最大値をρ_{max}とし、$-\rho_{max} \leqq \rho \leqq \rho_{max}$の範囲を分割した。この$\Delta\theta \times \Delta\phi \times \Delta\rho$の画素ごとに何本の曲線が通過したかを数え上げる。

(2) ３次元配列 $[\theta_i]$ $[\phi_j]$ $[\rho_k]$ で、そのまま処理すれば、膨大なメモリ容量を必要とするので、輿水が提唱しているＳＷＧＡ（Slit Window Guided Algorithm）[10][11]を３次元に拡張して、２次元配列 $[\phi_j]$ $[\rho_k]$ をθ方向にスライドさせる方法を採用した。この結果、２次元配列で処理が可能になり、細かいきざみ幅での高精度な平面抽出が可能になった。

(3) 各直線を３次元Ｈｏｕｇｈ変換して、曲線を３次元配列上に展開する場合、本来交わるべき直線は、必らず交わるように、６連結の点列で曲線を描くようにした。

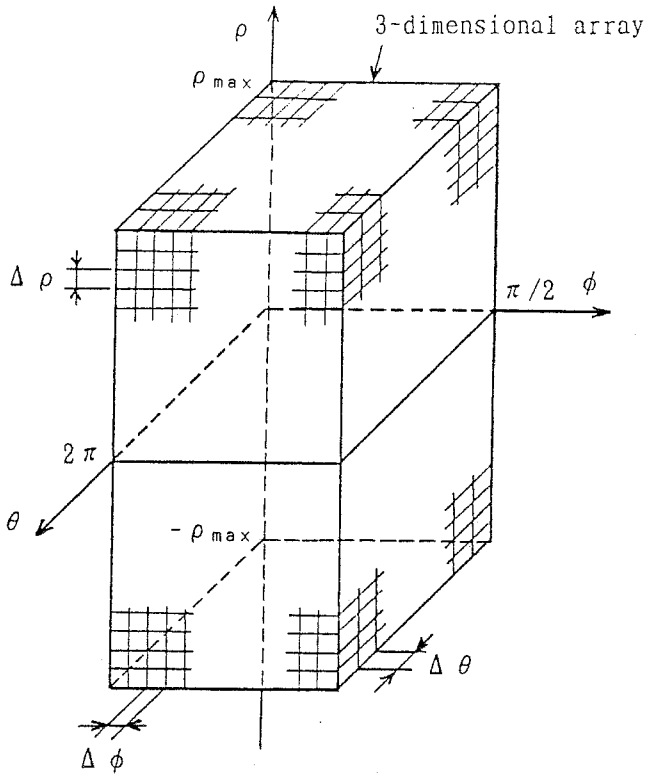


図４．７　３次元Ｈｏｕｇｈ空間の分割

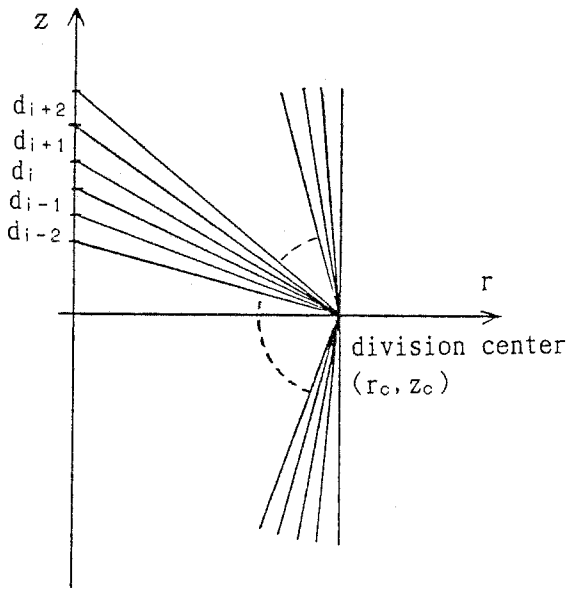


図４．８　ｚ軸の分割方法

（4）検出すべきピーク点は、３次元配列上の要素の値が、予め与えられたしきい値より大きく、かつ、３×３×３の近傍で周囲の26要素（または、５×５×５の近傍で周囲の124要素）のどの値に比べても大きいか等しいとき、検出することとした。

同様にアルゴリズム２をインプリメントしたときの留意事項は次のとおりである。

(1) ｚ切片の値で直線を分類するのに、ｚ軸の－∞から∞までの範囲を均等の間隔で分割するのは賢明ではない。そこで、図４．８に示すように、分割中心（r_c、z_c）を指定し、その内側（ｚ軸側）の角度（180[deg]）を均等の微小角度で分割し、分割する各直線がｚ軸と交わる交点のｚ座標ごとにｚ軸を分割した。この各分割区間をd_iで表わす。この結果、z_cに近いほど小さな分割間隔になり、離れるほど大きくなった。

(2) 各直線ごとにｚ切片の値を計算し、(1)の方法で求めたｚ軸上の区間d_iごとに直線を分類する。この区間d_iに属する直線の数が、予め与えた閾値以上であり、かつ、隣接する区間（d_{i-1}とd_{i+1}）に属する直線の数以上であるとき、d_{i-1}・d_i・d_{i+1}の３区間に属する直線は同一のｚ切片を持つものとして抽出した。この直線の部分集合に対するｚ切片の値は、中央の区間d_iの中央値で代表させた。

(3) (2)で抽出された直線の部分集合ごとに、（θ，ρ）の２次元空間上で曲線間の交点を求める。このときのρ方向の分割は、次の方法で行った。部分集合に含まれる直線ごとにとりうるρの範囲を計算し、それらのρの範囲の和の部分を求め、その区間を均等の間隔で分割した。この結果、与えられた分割数に対して、最小の分割幅を実現することができた。

(4) ２次元配列［θ_i］［ρ_k］上に曲線を展開するときは、交わるべき曲線は必ず交わるように、４連結の点列で曲線を描くようにした。

(5) 検出するピーク点は、２次元配列［θ_i］［ρ_k］上の要素の値が、予め与えた閾値より大きく、かつ、３×３の近傍で周囲の８要素のどの値に比べても大きいか等しいとき、検出することとした。

４．４　実験と検討

以上の方法でインプリメントしたプログラムを使用して、直線データの集合から平面の抽出を検討した。まず、４．４．１では、計算で求めた理想的な直線データの集合から平面抽出を行い、４．４．２では、実測した物体表面上の直線データの集合から平面抽出を行った。

４．４．１　理想的な直線データからの平面抽出

１辺６cmの正１２面体に対して、仮想的に回転テーブルを１[deg]回転させるごとに光切断法で計測したとして、直線データを計算し、これらの直線データの集合（720本）から平面抽出を試みた。計算で求めた平面パラメータは、$\hat{\theta}$ = 36.0、

108.0、180.0、252.0、324.0[deg]、$\hat{\phi}$ = 63.43[deg]、$\hat{\rho}$ = ±66.81[mm]である。

アルゴリズム1による結果を表4.1に、アルゴリズム2による結果を表4.2に示す。アルゴリズム1でもアルゴリズム2でも、上下の面を除く10個の平面を検出できた。平面の検出分解能の設定（3次元Ｈｏｕｇｈ空間の分割数）は同一であり、各パラメータとも検出分解能以下の誤差で検出できている。要した計算時間は、アルゴリズム1が466秒、アルゴリズム2が126秒であり、30%以下に短縮された（ＳＵＮ－３ワークステーションの場合）。

表4.1　アルゴリズム1による理想直線データからの平面抽出

No.	ピーク値	$\hat{\theta}$ [deg]	$\hat{\phi}$ [deg]	$\hat{\rho}$ [mm]
1	73	36.0	63.0	-67.0
2	73	36.0	63.0	67.0
3	73	108.0	63.0	-67.0
4	73	108.0	63.0	67.0
5	73	180.0	63.0	-67.0
6	73	180.0	63.0	67.0
7	73	252.0	63.0	-67.0
8	73	252.0	63.0	67.0
9	73	324.0	63.0	-67.0
10	73	324.0	63.0	67.0
11	なし			

$\Delta\theta$ = 1 [deg], $\Delta\phi$ = 1 [deg],
$\Delta\rho$ = 1 [mm] (-100[mm]$\leqq\rho\leqq$100[mm])

表4.2　アルゴリズム2による理想直線データからの平面抽出

No.	ピーク値	$\hat{\theta}$ [deg]	$\hat{\phi}$ [deg]	$\hat{\rho}$ [mm]
1	73	36.0	63.24	-66.76
2	73	108.0	63.24	-66.76
3	73	180.0	63.24	-66.76
4	73	252.0	63.24	-66.76
5	73	324.0	63.24	-66.76
6	73	36.0	63.24	-66.76
7	73	108.0	63.24	66.76
8	73	180.0	63.24	66.76
9	73	252.0	63.24	66.76
10	73	324.0	63.24	66.76
11	(16)			

z切片の分割中心　r_c = 100[mm], z_c = 0[mm]
z方向の分割数 180, $\Delta\theta$ = 1 [deg],
ρ方向の分割数 200

4．4．2　実測した直線データからの平面抽出

1辺約6cmの正12面体に対して、実際に回転テーブルを1[deg]回転させるごとに光切断法で計測し、得られた直線データの集合から平面抽出を試みた。

直線データの抽出は、光切断面の輪郭線データを折れ線で近似し、与えられた閾値以上の長さをもつ線分に対応する部分は直線データであるとした。さらに、その線分に対応する点の系列に対して、最小自乗法を使った直線方程式のフィッティング操作を行い、より精度の高い直線の方程式を算出した。

アルゴリズム1による結果を表4．3に、アルゴリズム2による結果を表4．4に示す。近似折れ線を構成する線分の内、10[mm]以上の長さをもつ線分を直線データとして抽出した。この結果得られた直線データは720本である。また、アルゴリズム1の結果は、5×5×5の近傍で極大となるピーク点のみを抽出した。平面の検出分解能の設定は、同一である。双方とも、余裕を持って上下の面を除く10個の平面を抽出できている（アルゴリズム1では、10番目の平面のピーク値が44、11番目はなしであった。同様に、アルゴリズム2では、10番目が60、11番目が16であった。）。平面抽出のための計算時間は、アルゴリズム1が463秒、アルゴリズム2が143秒であり、およそ30%に短縮された（SUN－3ワークステーションの場合）。

また、図4．9は、アルゴリズム2において、実測した直線データの集合をz切片の値で分類した1部分集合の直線データに対して、2次元Hough空間（θ, ρ）上でのHough曲線を描いたものである。3点で曲線が誤差なく交わっており、精度よく平面を抽出できることがわかる。

表4．3　アルゴリズム1による実測直線データからの平面抽出

No.	ピーク値	$\hat{\theta}$ [deg]	$\hat{\phi}$ [deg]	$\hat{\rho}$ [mm]
1	75	180.0	66.0	-76.0
2	73	252.0	65.0	-77.0
3	72	108.0	66.0	-77.0
4	72	323.5	64.0	-79.0
5	71	36.0	64.0	-79.0
6	70	179.0	64.0	54.0
7	68	324.0	62.0	51.0
8	64	108.0	64.0	53.0
9	51	37.0	62.0	51.0
10	44	252.0	63.0	52.0
11	なし			

$\Delta\theta = 1$ [deg], $\Delta\phi = 1$ [deg],
$\Delta\rho = 1$ [mm] (-100[mm]$\leqq\rho\leqq$100[mm])

表４．４　アルゴリズム２による実測直線データからの平面抽出

No.	ピーク値	$\hat{\theta}$ [deg]	$\hat{\phi}$ [deg]	$\hat{\rho}$ [mm]
1	74	180.0	65.13	-75.91
2	74	323.0	62.81	50.76
3	73	108.0	64.80	-76.84
4	73	252.0	64.80	-76.84
5	72	36.0	64.14	-78.70
6	72	323.0	64.14	-78.70
7	71	36.0	62.81	50.76
8	69	252.0	63.85	52.53
9	67	107.0	63.85	52.53
10	60	179.0	63.85	53.47
11	(16)			

z 切片の分割中心　r_c = 100[mm], z_c = 0[mm]
z 方向の分割数 180, $\Delta\theta$ = 1 [deg],
ρ 方向の分割数 200

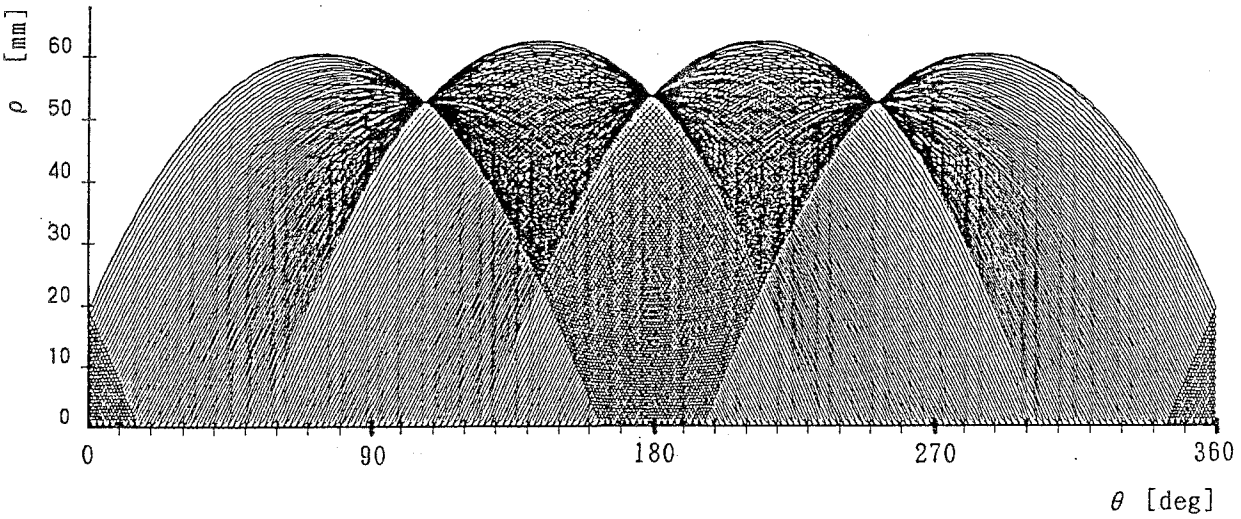


図４．９　（θ，ρ）空間のＨｏｕｇｈ曲線例

４．４．３　検討

３次元空間（ｘ，ｙ，ｚ）上の点は、３次元Ｈｏｕｇｈ空間（θ，ϕ，ρ）では曲面に対応し、同じく、３次元空間上の平面は、Ｈｏｕｇｈ空間では点に対応する。３次元空間上の直線は、点と平面の中間的なものと考えることができ、Ｈｏｕｇｈ空間では曲線に対応する。このようにＨｏｕｇｈ空間上に展開される曲面と曲線の違いから、平面抽出に必要な計算量を低減させることができた。

３次元空間上に多数の曲線があって、その交点を求める問題の場合、ここでは実際に３次元空間上に曲線を描き、その曲線間の交点を求めるという直接的な方法で行った。このような問題を解く場合、まず、３次元空間上の曲線を２次元に投影した形で描いてその曲線間の交点を求め、そのあと、２次元上のその交点が

もう１軸方向について交点を構成するかどうかをチェックする方法も考えられる。この方法によれば、３次元空間上での問題が、２次元＋１次元の問題に分割され、そのために必要な計算量は、飛躍的に軽減される。理論的には、３次元空間上で直接交点を求める必要があるが、Ｈｏｕｇｈ空間上の交点をもとめるような性質の問題では、後者の方法をとっても交点の検出精度がそれほど落ちない可能性があり、かつ、計算量は著しく低減されると予想される。この方法の検討は今後の課題である。

筆者らの形状入力システムで得られる直線データには、もし同一平面上に存在すれば、必ず同一のｚ切片をもつという性質がある。本報告では、Ｈｏｕｇｈ変換で直接平面のパラメータが抽出できるように、（θ，ϕ，ρ）のＨｏｕｇｈ空間上で議論した。一つの別手法として、ｚ切片の値で分類した直線の部分集合で、各直線とｘ－ｙ平面との交点が一直線上に並ぶかどうかをチェックしても可能である。いずれの方法にしても、ｚ切片の値で分類しておけば、その後の処理は、２次元の問題に帰着される。ただ、ｘ－ｙ平面上の交点の点列が一直線上に並ぶかどうかのチェックは、規則的な点列であるため、その点列に対して折れ線近似を行ったのち、最小自乗法による直線方程式のフィッティングを行うことにより可能である。この方法によれば、Ｈｏｕｇｈ変換を用いなくても平面抽出が可能であり、そのために必要な計算量は、極端に少なくなる。したがって、筆者らの形状入力システムで得られる直線データの集合から平面を抽出する場合、計算量に関する限り、ここで検討した方法よりは、Ｈｏｕｇｈ変換を用いないこの手法が、より適当であると考えられ、この方法との比較・検討は今後の課題である。

5．おわりに

本報告では、光切断法を用いた立体形状入力について、(1) 計測データを折れ線近似して三角パッチの節点を発生させる方法、(2) 形状データの欠落部分を補完して完全な立体形状を生成する方法、(3) 3次元Ｈｏｕｇｈ変換を使って物体表面上の平面を抽出する方法を中心に検討結果を述べた。

三角パッチによる立体形状の再構成方法では、節点を折れ線近似により発生させたため、原形状を特徴づける稜線や頂点等が保存され、かつ、データ量の圧縮も実現された。ただ、この方法では、$\Delta\theta$回転させるごとの計測データ上に発生させた節点をすべて用いるため、半径ｒの値により三角パッチの生成密度が異なるという問題点は残っている。回転方向も考慮した、さらに効率的で知的な立体形状の再構成方法の検討は、今後の課題である。

欠落した形状データの補完方法については、回転テーブルにより物体を回転させて、物体の全周囲をスリット光の方法に向けるのは比較的簡単であるが、上部・下部等をスリット光の方向に自動的に向けるのは難しい。そこで、回転テーブル上への物体の置き方を変えて入力した形状データから、全体形状を合成するという方法を採った。このアイデアは、(株)ＡＴＲ通信システム研究所山下紘一社長の御提案によるものであり、深く感謝する。本研究では、一方の形状データで欠落した部分を他方の形状データで補完する手法を提案した。この手法は、本形状入力方法の特徴をうまく利用し、かつ、少ない計算量で全体形状を生成するものである。本手法をウサギの置き物に適用し、欠落部分のない全体形状を忠実に再構成した。ただ、現在のインプリメントでは、２形状データの位置合わせは人手の指示に頼っており、自動化されていない。この位置合わせの自動化についても若干の検討を行った。ここで提案した手法では、予め大まかな位置合わせができていれば、最適な位置に収束することが確認された。今後の課題としては、大まかな位置合わせの方法、最適位置への収束の加速方法等が考えられる。このほか、形状合成処理の課題として、(1) 単に欠落部分を補完するのでなく、部分部分により双方の形状データのどちらを採用するかを検討し、三角パッチの生成密度の均等化を図ること、(2) 貼り合わせる部分に段差が生じたときでも、滑らかに貼り合わせる方法の確立等が挙げられる。

３次元Ｈｏｕｇｈ変換による平面抽出方法では、直線を直接３次元Ｈｏｕｇｈ変換するという方法により、Ｈｏｕｇｈ変換による平面抽出に必要な計算コストを実用的なレベルにまで低減させた。Ｈｏｕｇｈ変換による手法は、一般に膨大な計算量を必要とする欠点はあるが、局所的に見つけ出した特徴量を広げていくのではなく、全体的に見渡して特徴量を抽出するという長所がある。本報告では、

直線を直接Ｈｏｕｇｈ変換することにより生成されるＨｏｕｇｈ曲線間の交点から平面を抽出する手法を提案した。また、筆者らが行っている光切断法による立体形状入力システムで得られる直線データの集合からの平面抽出を検討し、その直線データの規則性を利用して、２次元Ｈｏｕｇｈ空間上でのＨｏｕｇｈ曲線間の交点から平面抽出を行うアルゴリズムを考案し、かつ、その有効性を評価した。今後の課題として、(1) ２次元Ｈｏｕｇｈ変換で開発されている各種の計算効率化手法の適用、(2) 平面の方程式だけでなく、平面の範囲を効率よく算出する方法の確立、(3) Ｈｏｕｇｈ変換を用いない平面抽出手法との比較検討等が挙げられる。

謝　辞

最後に、本研究を進めるにあたり、いろいろと御指導頂いた（株）ＡＴＲ通信システム研究所葉原耕平会長、有益な御提案を頂いた山下紘一社長に深く感謝致します。また、有益な御助言・御討論を頂いた知能処理研究室Daniel T. L. Lee主幹研究員、肥塚隆研究員、田中弘美客員研究員はじめ関係諸氏、並びに、ソフトウェアの作成・実験に御協力頂いた浦真吾氏に感謝します。

文　献

(1) 青山，河越，佐藤：”曲率に大きな幅を持つ線図形の直線近似”，第32回情処全大，7N-8(1986)

(2) 安居院，斉藤，中島：”背面鏡を用いた３次元物体データ入力システム”，信学論(D)，J70-D，5，pp.995-1002(1987)

(3) L.D.Floriani,B.Falcidieno,C.Pienovi:"Delaunay-based Representation of Surfaces Defined over Arbitrarily Shaped Domains", CVGIP, 32, pp.127-140(1985)

(4) 池谷，吉田，小寺：”光切断法三次元計測の高精度化について”，昭63信学春季全大,D-245(1988)

(5) 井口征士：”三次元計測研究　最近の動向と展望”，映像情報，18，11(1986)

(6) 伊理，腰塚：”計算幾何学と地理情報処理”，ｂｉｔ，1986年９月号別冊，共立出版(1986)

(7) R.A.Jarvis: "A Perspective on Range Finding Techniques for Computer Vision",IEEE Trans. Pattern Anal. Machine Intell., PAMI-5,2,pp.122-139(1983)

(8) 情報処理振興事業協会："SPIDER-II Users' Manual"(1986)

(9) 肥塚，秋山，小林：”モアレ法による３次元立体形状入力に関する一考察”，信学技報,PRU87-40，pp.17-22(1987)

(10) 輿水大和："直線パターン検出のためのＨｏｕｇｈ曲線追跡型アルゴリズムについて",信学論(D)，J68-D,10,pp.1769-1776(1985)

(11) 輿水,村上："直線群検出のためのＨｏｕｇｈ曲線追跡型アルゴリズム"，信学論(D)，J69-D,4，pp.631-633(1986)

(12) 輿水大和："Ｈｏｕｇｈ変換に関する最近の研究動向",情処研報，87-CV-51-1(1987)

(13) M.J.Magee, J.K.Aggarwal:"Using Multisensory Images to Derive the Structure of Three-Dimensional Objects - A Review", CVGIP, 32, pp.145-157(1985)

(14) 三輪，片山：”山登り法による画像の貼り合わせ”,第35回情処全大，1J-2(1987)

(15) 中沢,大矢,中島,油田："ファイバーグレイティングを用いた３次元形状計測における要素平面群の抽出法",第17回画像工学コンファレンス，3-8(1986)

(16) 成瀬，野村，山本：”スリット光投影法による高精度距離・姿勢計測”，信学論(D)，J69-D，12，pp.1888-1894(1986)

(17) 西川, 三宮, 茨木：" 岩波講座情報化学－19 最適化", 岩波書店(1982)

(18) H. Nishino, K. Akiyama, Y. Kobayashi:"Consideration on Automatic Acquisition and Reconstruction of an Object Shape", Proceedings of the Second International Workshop on Signal Processing of HDTV, pp. 295-302(1988)

(19) 西野, 秋山, 小林：" 光切断法による３次元立体形状自動入力に関する一考察", テレビ学技報, 11, 16(1987)

(20) 西野, 秋山, 小林：" 光切断法による３次元立体形状自動入力に関する一検討",第36回情処全大, 5W-7(1988)

(21) 西野, 肥塚, 秋山, 小林：" 光切断法による３次元立体形状入力と形状合成", テレビ学技報, 13, 6(1989)

(22) 西野, 秋山, 小林：" ３次元Ｈｏｕｇｈ変換による直線集合からの平面抽出", 第38回情処全大, 5C-5(1989)

(23) 西野, 肥塚, 秋山, 小林：" ３次元立体形状の入力と再構成に関する一検討", 1989年信学春季全大, SD-3-5(1989)

(24) 大矢,中沢,中島,油田："ファイバーグレイティング視覚センサの開発－Ｈｏｕｇｈ変換を用いた平面の検出法－",昭62信学総全大,1586(1987)

(25) F. Schmitt, H. Maitre, A. Clainchard, J. Lopez-Krahe: "Acquisition and Representation of Object Surface Data",2-nd International Technical Symposium on Optical and Electro-Optical Applied Science and Engineering(1985)

(26) 崔,安居院,中島："特徴保存型階層化高速ハフ変換を用いたナンバプレート領域抽出", 信学論(D), J71-D, 5, pp. 856-862(1988)

(27) K. Wall, P. E. Danielsson, "A Fast Sequential Method for Polygonal Approximation of Digitized Curves", CVGIP, 28, pp. 220-227(1984)

(28) C. K. Wu, D. Q. Wang, R. K. Bajcsy: "Acquiring 3-D Spatial Data of a Real Object", CVGIP, 28, pp. 126-133(1984)

(29) 山本, 田宗, 田村：" 距離画像の入力と処理", 信学技報, PRU86-129 (1987)

付録Ａ．作成ソフトウェアの説明

Ａ．１　ソフトウェアの一覧

本研究を行うために作成したソフトウェアの一覧は、次のとおりである。

表Ａ．１　ソフトウェアの概略機能

No	名　称	概　略　機　能	作成者
1	henkan	カメラ変換マトリクスＡを作成するのに必要なデータ（３次元座標(x,y,z)とカメラ座標(u,v)の対応表）を作成する。（２．１．４参照）	浦
2	slit7	回転テーブルを利用して物体を回転させながら全周囲の形状を自動的に計測し、三角パッチの集合でサーフェイスモデルを再構成する。出力は、.cooファイル（バイナリー）と.objファイル（アスキー、WAVEFRONTで読込可）である。（第２章参照）	浦 西野
3	grahole2	slit7 で入力した形状データ（.objファイル）における欠落部分を検出する。（３．２．１参照）	浦
4	gracmpl	メイン・サブの形状データの位置合わせを行い（手動）、座標変換マトリクスを出力する。（３．２．２参照）	浦
5	mergel	メインの欠落部分内部にサブの形状データを使って三角パッチを生成する形状補完処理を行う。（３．２．３参照）	西野
6	graautol	山登り法を使って、メインとサブの形状データの位置合わせを自動的に行う。（予め、大まかな位置合わせを行っておく必要がある。３．４参照）	浦
7	coowav	.cooファイル（バイナリー）から.objファイル（アスキー）を作成する。	浦
8	hough1	直線を直接Ｈｏｕｇｈ変換して、３次元Ｈｏｕｇｈ空間上にＨｏｕｇｈ曲線描き、そのピーク点から平面を抽出する。（４．２．２参照）	浦
9	hough2	直線をｚ切片の値で分類したのち、２次元Ｈｏｕｇｈ空間上のＨｏｕｇｈ曲線のピーク点から、平面を抽出する。（４．２．３参照）	西野

以上のプログラムのうち、ｇｒａｃｍｐ１以外は、ＶＩＣＯＭ／ＶＭＥ（ＣＶ０１）上で動作し、ｇｒａｃｍｐ１は、ＩＲＩＳ－3030（ＣＩＲＩＳ０１）上で動作する。ＶＩＣＯＭ上で動作するプログラムのうち、ｇｒａｈｏｌｅ２、ｍｅｒｇｅ１、ｃｏｏｗａｖ、ｈｏｕｇｈ１、ｈｏｕｇｈ２は、ＶＩＣＯＭの画像処理機能を使用しておらず、通常のＳＵＮ３ワークステーションの環境下で動作する。

ここでは、各ソフトウェアの概略機能を述べるにとどめ、詳細な使用方法等は、ソフトウェア取扱説明書において説明する。

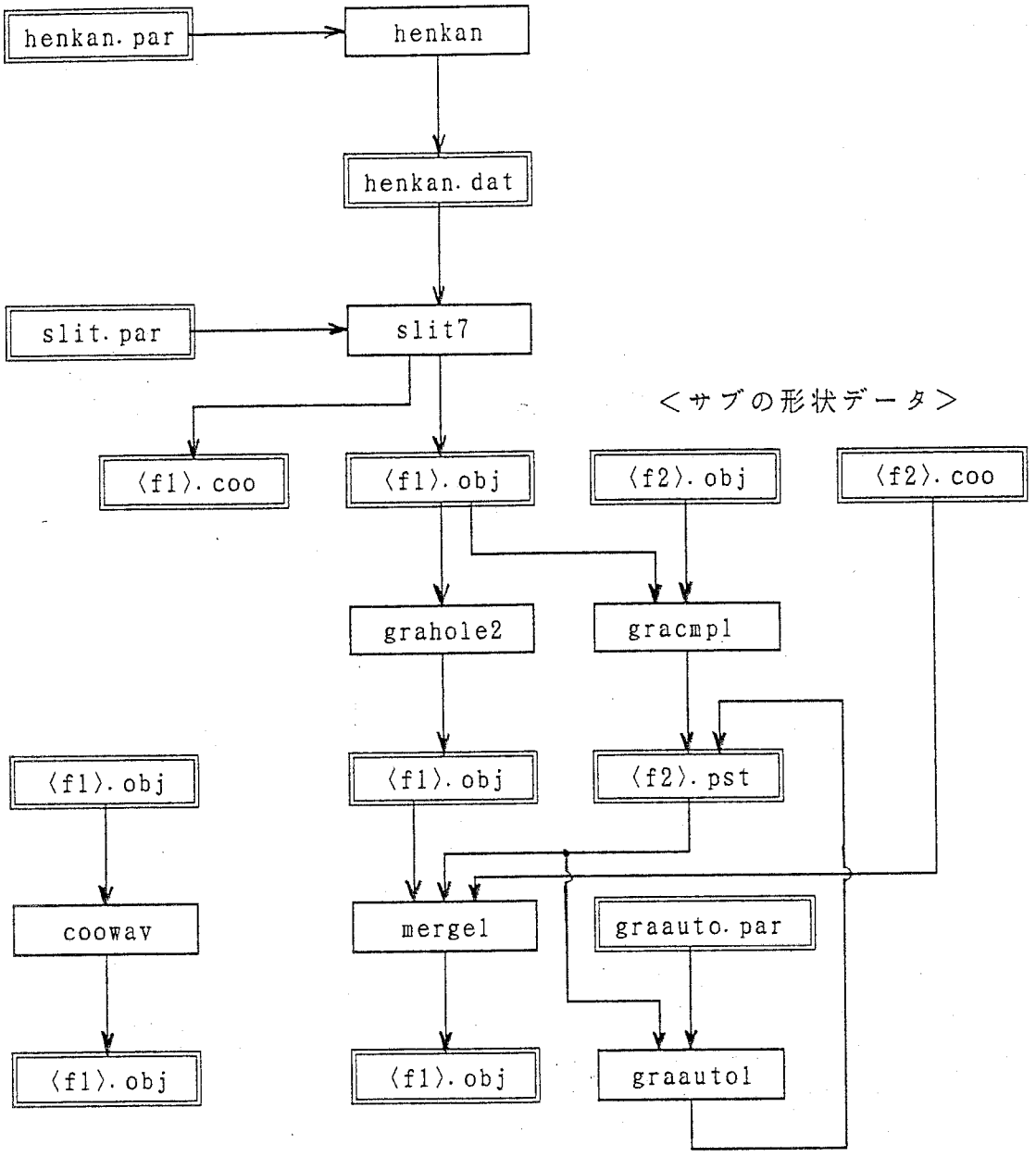
表Ａ．１のほかに、光切断法による本形状入力システムの関連で作成した西田氏仕様のプログラムは、次のとおりである。

表Ａ．２　その他のソフトウェアの概略機能

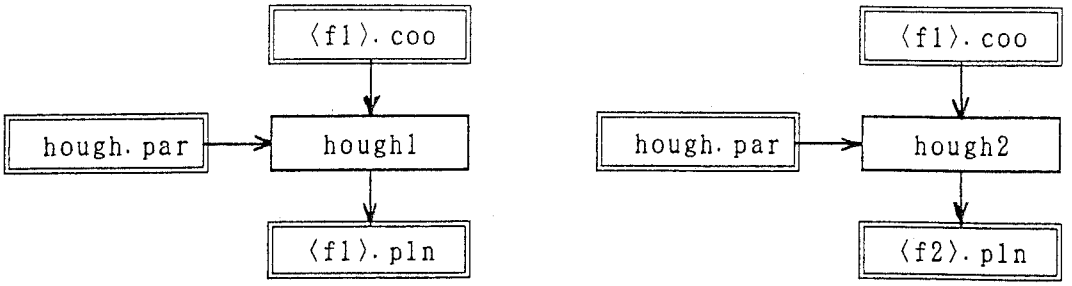
No	名　称	概　略　機　能	作成者
1	grasmp	slit7 で出力されたファイル（.coo）から、ｚ軸方向に一定の間隔でサンプリングした３次元データ（円柱座標系）を生成し、バイナリー形式でファイル（.smp）に出力する。	浦
2	smpwav	.smp ファイルから .obj ファイル（WAVEFRONT で読込可）を生成する。	浦
3	smparea	.smp ファイルから、入力した立体形状をあるｚ座標で切断したときの断面図をＳＵＮのディスプレイ上に表示する。	大内
4	smpint	.smp ファイルから、（θ，ｚ）に対する半径ｒの値の分布状況を、濃淡画像としてＶＩＣＯＭに表示する。	大内

A．２　全体的な処理の流れ

表Ａ．１の各プログラムについて、全体的な処理の流れを説明する。本文の第２章・第３章の関係を図Ａ．１に、第４章の関係を図Ａ．２に示す。



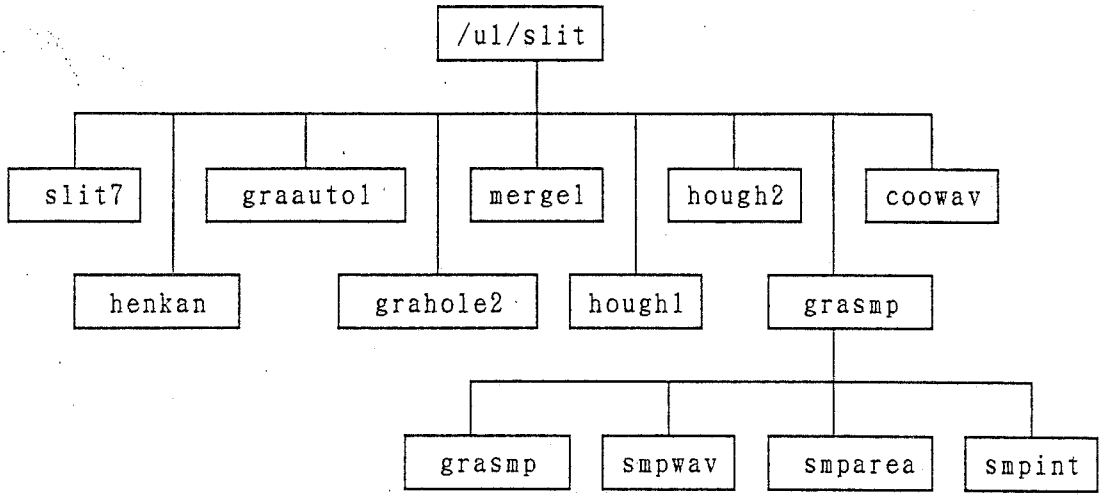
図Ａ．１　形状入力・再構成・形状補完に関する処理の流れ



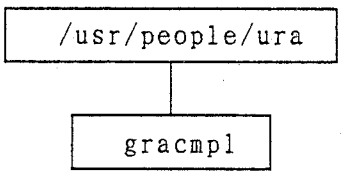
図A．2　平面抽出に関する処理の流れ

A．3　ディレクトリ構造

ＶＩＣＯＭ／ＶＭＥ（ＣＶ０１）上のディレクトリ構造を図A．3に、ＩＲＩＳ-3030（ＣＩＲＩＳ０１）上のディレクトリ構造を図A．4示す。このディレクトリ構造の下にも階層的にサブディレクトリが存在するが、詳細はソフトウェア取扱説明書において説明する。



図A．3　ＶＩＣＯＭ上のディレクトリ構造



図A．4　ＩＲＩＳ上のディレクトリ構造

付録Ｂ．形状入力の手順

Ｂ．１　ＶＩＣＯＭの起動・終了方法

ＶＩＣＯＭ／ＶＭＥ（ＣＶ０１）の起動・終了方法は、次のとおりである。

起動方法

(1) ＳＵＮ・ＶＩＣＯＭのモニタ等周辺装置の電源を入れる。
(2) ＶＩＣＯＭ本体背面のブレーカを入れる。
(3) ＶＩＤＩＳ前面の電源を入れる。
(4) ＶＩＣＯＭ本体前面のキーを"on"に回す。
(5) ＶＩＣＯＭ本体前面の電源スイッチを押す。
(6) ＶＩＣＯＭ本体前面のキーを"lock"に回す。
(7) ＳＵＮコンソールに"login"のプロンプトが表示されて、起動完了。

終了方法

(1) ユーザ名"owari"でログインする。（コンソールに")"のプロンプトが表示される。）
(2) 周辺装置の電源を切る。
(3) ＶＩＤＩＳ前面の電源を切る。
(4) ＶＩＣＯＭ本体前面のキーを"on"に回す。
(5) ＶＩＣＯＭ本体前面の電源スイッチを切る。
(6) ＶＩＣＯＭ本体前面のキーを"off"に回す。
(7) ＶＩＣＯＭ本体背面のブレーカを切る。

Ｂ．２　形状入力の操作手順

第２章で述べた手法により、実際に形状入力を行う際の手順を説明する。

(1) ＴＶカメラの設置とケーブルの接続

ＴＶカメラは、ＶＩＣＯＭと同じ同期信号で動作させ、回転テーブルに向かって右側のカメラの出力信号をＶＩＣＯＭ 512 系の３チャンネル（Ｒ・Ｇ・Ｂ）入力のＲチャンネルに、左側のカメラの出力信号をＧチャンネルに接続する。本来は、白黒カメラを使うのがよいが、現在はカラーカメラを使っているため、カラーＮＴＳＣ信号をそのままＶＩＣＯＭの各チャンネル（Ｒ・Ｇ）に入力する。そのため、カラー成分による若干のノイズが発生する。ＴＶカメラの位置は。通常、スリット光に対して約30[deg]の角度をおいて設置し、回転テーブル上の物体が全

部入るように、ズーム・ピント・カメラ位置等を調整する。いったんＴＶカメラの調整を行えば、これ以後、調整を変化させてはならない。

(2) スリット光の調整

回転テーブル上の物体を取り除き、キャリブレーション板（図２．６）をスライドプロジェクタに対してほぼ直角になるように設置する。スライドプロジェクタにスリット光用スライド（幅が、5 ﾐｸﾛﾝ と10 ﾐｸﾛﾝ のガラススライドがある）を挿入し、点灯する。スリット光の位置を変える（ハンドルを回転させてスライドプロジェクタを移動させるとともに、スライドの水平度を調整する）ことにより、スリット光がちょうど回転テーブルの回転軸を通過するように調整する。このあと、スライドプロジェクタが移動しないように固定する。

(3) ３次元座標とカメラ座標の対応づけ（キャリブレーション）

キャリブレーション板上の数カ所にマークを貼る（図２．７参照）。このマークは、回転テーブルを回転させても、ＴＶカメラで観測可能な位置に貼る。このマークの３次元座標をもとにキャリブレーションを行う。次に、回転テーブルを約90[deg]回転させて、スリット光の平面とキャリブレーション板が同一平面になるように調整する（目視で確認する）。この位置から再び回転テーブルをちょうど90[deg]回転させて、その位置を基準位置とする。すなわち、スリット光平面がｙ－ｚ平面、キャリブレーション板がｘ－ｙ平面になるとする（本文では、回転テーブルの回転軸をｚ軸として述べているが、ここでは、WAVEFRONT の呼称方法に合わせて、回転軸をｙ軸とする）。マークを貼った板上の位置（２次元）、及び、回転テーブルを回転させる角度等をパラメータファイル（henkan.par）に登録する（エディットで入力する）。この指定方法は、ソフトウェア取扱説明書で詳しく説明する。このあとプログラムｈｅｎｋａｎを実行する。ＶＩＣＯＭのモニタ上に、ＴＶカメラで入力されたキャリブレーション板が表示されるので、その板上のマーク位置をカーソルで順々に入力する（マウスの左ボタン）。その順番は、ＳＵＮのモニタ上に表示される。なお、キャンセルはマウスの中ボタン、その画像での入力終了はマウスの右ボタンを押したのち、ＳＵＮのキーボードでリターンを入力する。この結果、回転テーブル上の３次元座標（ｘ，ｙ，ｚ）とＴＶカメラの画像上の座標（ｕ，ｖ）の対応表をファイル（henkan.dat）形式で出力する。

(4) ｓｌｉｔ７の実行

キャリブレーション板を撤去し、入力対象物体を回転テーブルの上に載せる。

このとき、回転テーブルを回転させて、物体の中心がほぼ回転軸に一致することを確認する。ここで、消灯する。次に、ｓｌｉｔ７を実行するのに必要な各種のパラメータをファイル（slit.par）に登録する（エディタで入力する）。なお、２値化処理のためのパラメータ（本文２．１．３で説明した閾値L_{th}）は、ＶＩＣＯＭのｃｉｍでカメラ画像を入力（dig）し、その画像を２値化（#thr）して、ノイズ成分が出てこない程度でできるだけ低く設定する。このあと、ｓｌｉｔ７を実行する。通常、１回転角度での処理時間は３～４秒程度である。回転テーブルが360[deg]回転したのち、三角パッチによる再構成処理が行われ、バイナリー形式のファイル（filename.coo）と WAVEFRONT でそのまま処理ができるアスキー形式のファイル（filename.obj）が出力される。

(5) WAVEFRONT による表示

ｓｌｉｔ７で生成されたファイル（filename.obj）をＦＴＰを使ってＩＲＩＳ－3030 に転送する。ＩＲＩＳ上で WAVEFRONT を起動し、MODEL を使ったワイヤフレームの表示等、各種の処理を実行できる。

以上が、本文第２章で説明した形状計測と再構成部分の実行手順の概要である。個々のプログラムを実行するときの詳細な方法は、ソフトウェア取扱説明書で説明する。