

ステレオ法による多面体の再構成

丹羽敏行 渡辺凡夫

名古屋工業大学工学部電気情報工学科

近年、コンピュータに人間の持つような視覚機能を与えようとする研究が、人工知能の1分野として研究されている。これらはコンピュータ・ビジョンと呼ばれ、2次元画像から3次元世界を再構成することが1つの研究目標である。

この再構成には、大きく分けて2つのアプローチがある。1つは、ある視点から撮られた1枚の写真から、もとの3次元世界を再構成する方法で、単眼視処理と呼ばれる。他のアプローチは我々に目が2つあるように、2あるいはそれ以上の視点から撮られた複数枚の画像から3次元世界を認識しようとするものである。これは、複眼視処理と呼ばれており、2台のカメラを用いて、その両眼視差から3次元位置を計測するステレオ法は、良く研究されてきた。ただ2つの画像間の対応付けをいかに行うかが課題となっている。

そこで、我々も、対象を多面体に限定し、その形状パラメータの計測とそれに基づく再構成を試みてきた。

今回より信頼性の高い頂点の検出と、頂点間での対応付けによる高速化を計ったので、その概要を報告する。

Reconstruction of Polyhedric Object by Stereo Method

Toshiyuki NIWA and Tsuneo WATANABE

Nagoya Institute of Technology
department of Electrical Computer engineering
Gokiso Shouwa Nagoya 466 Japan

In these days, the research that giving computers visual recognition facility, just like a human, is making progress as one section of A.I. It's called Computer Vision. The reconstruction of 3-Dworld from 2-Dimage is one of its theme for us. In general, there are two approaches to reconstruct 3-Dworld.

One is called Monolithic Vision, it makes use of only single clairvoyance picture. And the other is Stereo Vision, it makes use of two or more than two pictures which have different view points and its method is considered because human has two eyes. This Stereo Vision is very famous and its method is that derives 3-Dworld coordinate position by difference of the same object on each scenes which are looked at human two eyes. The only important thing is how should we do pattern matching of these vertices. Then we too, experiment on construction of 3-Dworld basing to measure shape parameters of object confined to Polyhedric.

In this experiment, we take care of two points below.

- (1) more reliable processing that is how to detect vertices
- (2) the speed up by relation of each vertices

and we show our trial in this report.

1 はじめに

ステレオ法では通常、2個のカメラを用いて、その両眼視差から3次元位置を計測する。しかし、カメラが2台になると、費用が増し、取り扱いが煩雑になるばかりでなく、なによりも計算量と誤差の増大を招くことになる。そこで、図1に示すように、カメラは1台で固定し、カメラ前方に設定されたターンテーブルを回転させて、複数の画像を入力した。この結果、費用はもとより、カメラ・パラメータが簡単になり、計算量が減少できた。

今回、図形の直線抽出にはHough変換を採用したが、大規模な記憶域を必要とする欠点を解消するため、1次元累積レイタ行列によるHough変換を試行してみた。

また、悪状況での多面体の画像撮影においては、しばしばエッジの欠損が生じ、その結果、不自然な2次元図形を得ることがある。それに対し、よく知られているHuffmanの辺線と頂点の線画パターンに拘束条件を付加わって、2次元図形補完を行った。

対応付けは頂点間で行うことにより、処理の高速化を計った。

扱う画像はカラー画像としたので、各画像につきR, G, B 3回の処理を要するが、多面体の各面に彩色したり、色のついた照明を用いることにより、エッジの検出を容易にすることも可能とした。尚、これら一連の処理(図2)は全てパソコン上で処理した。

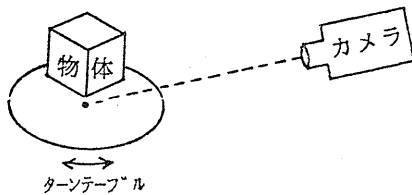


図1 概念図

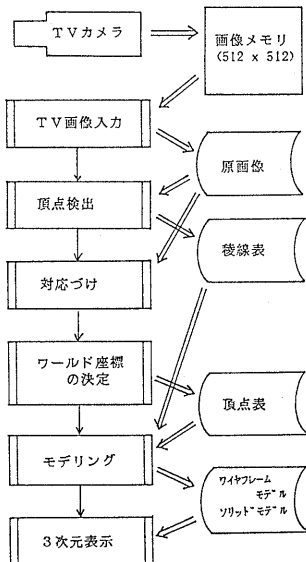


図2 処理概要図

2 育ち処理

入力された画像からエッジ検出を容易にするための雑音除去、画像強調処理を行う。対象はR, G, Bを基調としたカラー画像なので微分処理までは1枚の画像につき3プレーン分の処理を逐次施し、それらを再び1枚の画像に構成する処理を経て、非極大点抑制等を行う。

2.1 画像の平滑化

平滑化処理による雑音の除去法は、近傍画素の平均濃度を採用すると、2つの領域の境界付近に中間的な濃度の部分が生じて、2つの領域の境界(エッジ)がぼけてしまう。そこで、エッジをぼかすことのないように辺線を含まない平滑化領域を図3のように9箇所定め、これらの中で濃度の分散が最小になる領域を平滑化領域とした。

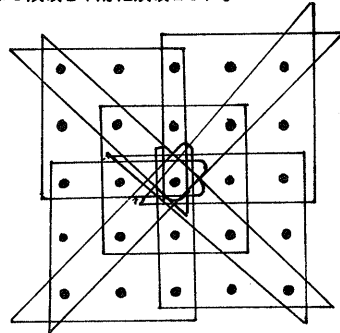


図3 平滑化領域

2.2 強調処理(エッジの検出)

画像の分割のための重要な方法は、濃度やテクスチャーが急激に変化している、1つの領域が終り他の領域が始まっている様な不連続性(エッジ)を検出する。濃度の急激な変化に伴って起こる線は、その両側の領域とは性質が異なった細い帯である。こうしたエッジを検出するために画像の微分を行うが、扱う画像がデジタル画像なので微分の代りに差分を用いる。

よく知られているデジタル勾配の近似にRobertsによる方法があり、今回はこの"Robertsの勾配"による近似を用いた微分処理を行った。

画像の勾配は任意の互いに直交する2方向とも隣接する2画素間の濃度差分から求められ、次に示すような斜め方向の4画素を利用して画像の勾配の大きさ $M(i, j)$ を次式から求める。 $M(i, j)$ が64階調を越す場合は64を、アンダー・シュートの場合はその絶対値とする。

$$M(i, j) = \{ \{ (f(i, j) - f(i+1, j+1))^2 + (f(i, j+1) - f(i+1, j))^2 \} / 2 \} \quad (1)$$

勾配の大きさ $M(i, j)$ は画像 $f(i, j)$ において辺線付近で大きな値となり、濃度がほぼ一定の位置では小さな値になる。したがって、濃度画像に対して差分処理を行って得られる画像 $M(i, j)$ は辺線が強調された画像となる。

1枚のカラー画像はR, G, B 3プレーンで構成されているので、微分画像を1枚のカラー画像に再構成する必要がある。今回は各プレーン6ビットの64階調を1バイトで処理を行ったため、各濃度値を計算速度とデータのコンパクト性を考慮して、R, G, BのMAXIMUMを再構成の際の濃度値とした。図4に微分画像と再構成画像を示す。Rで検出されなかったエッジがBやGで検出されていることに注目されたい。

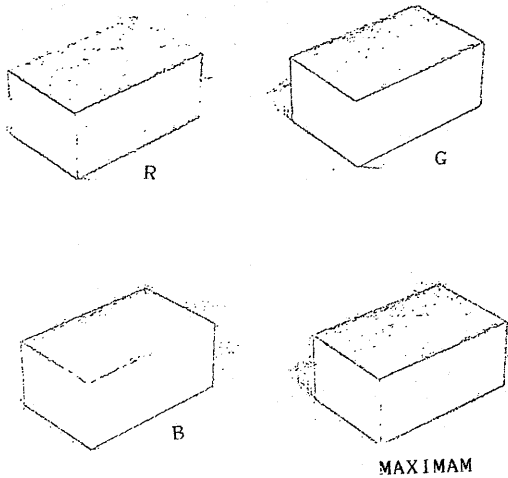


図4 微分画像

先の微分処理により、図形のエッジが検出されるわけであるが、微分処理だけの画像では、しきい値処理を行うと1本のエッジが2箇所以上で検出されることがあり、結果としてエッジが太くなり、直線の抽出が困難となる。そこで、エッジの水平、垂直方向についてスキャンラインを作り、現行の点の濃度値D(P)とスキャンライン上で1点後ろの濃度値D(P-1)を比較して、D(P)がD(P-1)よりも大きければ点(P-1)を削除する。こうして水平、垂直方向についての極大点のみを残してゆき、微分画像のつながりを明確に細線化する。

次に、非極大点の抑制を施した画像に対して今度はしきい値処理を行う。つまり、雑音と思われるような点を削除してしまい、次のHough変換での無駄をなくし、精度の良い直線を得るためである。会話形式により何回でもしきい値の設定を可能にし、エッジよりも高い濃度値を持つ雑音除去も可能にした。

非極大点の抑制及びしきい値処理画像を図5に示すので参照されたい。

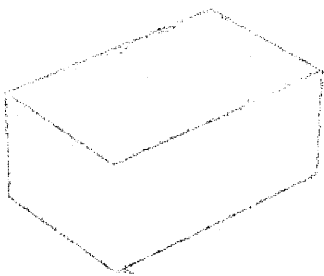


図5 非極大点抑制画像

3 画像の直線抽出

3.1 Hough変換

画像処理でよく使われる直線のあてはめ(直線近似)にHough変換による方法がある。これは図6に示すように、幾つかの点が直線(2)上に存在していると仮定し、直線(3)から(a, b)を変数とみなした(a, b)平面上での直線(4)を考える。

$$y = a \cdot x + b \quad (2)$$

$$y_i = a \cdot x_i + b \quad (i = 1, 2, \dots, M) \quad (3)$$

$$b = -x \cdot a + y \quad (4)$$

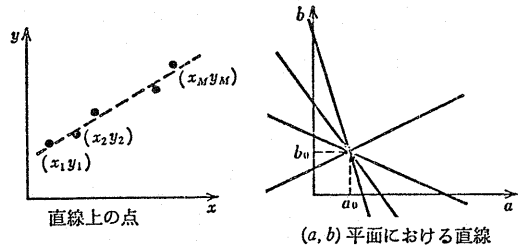


図6

図7

(x_i, y_i) が全て同一直線上の点、という仮定より、直線(4)は(x, y)のいかにかわからず、(a, b)平面上の点(a, b)を集約的に通過する。(図7参照)そこで(a, b)平面を十分細かくデジタルに区切り、(x_i, y_i)に対応している直線が通過する全てのメッシュ(a_n, b_n)に1ずつ累積してゆく。これを画像上の全ての(x, y)について行い、累積値が極大となるメッシュ(a_0, b_0)を求める直線のパラメータと決定できる。一般には(a, b)平面上への変換では、 a, b の十分に大きな所までメッシュを区切らねばならないという物理的困難な制約のために、極座標へ変換する以下の式を用いる。

$$\rho = x \cdot \cos \theta + y \cdot \sin \theta \quad (5)$$

ρ と θ を新しい変数と考えた場合、上式は点(x, y)を通るように制約された直線のパラメータ間の関係を与え、上式は(ρ, θ)空間における正弦曲線に対応する(図8)

画像平面における同一直線上の点の集合について、上式によって(ρ, θ)空間で決定された曲線は(画像点を通る直線の実際のパラメータに対応する)1つの共通点ですべて交わらねばならない。($(\rho-\theta)$ 空間の(ρ_0, θ_0))

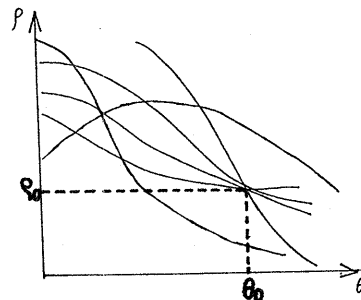


図8 ($\rho-\theta$)空間

3.2 最小二乗法の適用

このように同一直線上の点のクラスタを検出するため、その直線の各点について (ρ, θ) 空間で変換曲線を作成し、3つあるいはそれ以上のこのような曲線同時発生に対応する点(交点)を選ばようにする。

デジタル計算機でこれを実行するため、 ρ, θ パラメータを量子化する。 θ は $0, 2\pi$ の間で変化し、 ρ は画像平面の大きさまでで十分である。(a-b) 平面の時と同様、 (ρ, θ) 空間における曲線の作成は曲線に沿う各座標点のカウントを増やすことに対応し、曲線の同時発生点を見つけるには、高いカウントの座標点を選べばよい。

性能は選んだ量子化に影響される。粗いデジタル化は近接している線を区別を困難にし、非常に細かい分解能では、典雅直線からほんの少し外れても、それを同一直線とみなさなくなる。

故に、画像により適当な量子化レベルを選定する訳だが、粗いメッシュを選定しても必要とする記憶域がかなり大規模になり、メモリの不足が生じる。そこで、1次元配列による Hough 変換を行い、メモリ節約を試みた。

3.2 1次元 Hough 変換

$(\rho-\theta)$ 空間のために要する配列はかなりの規模になるが、図8を見てもわかるように $(x-y)$ 座標上の点の軌跡が $(\rho-\theta)$ 空間において占める割合は極めて小さい。またその軌跡の中でも必要とされるのは画像のドット数分の軌跡が周期的に集中している特定の (ρ_0, θ_0) 座標であり、多少のずれはあるものの量子化された軌跡の交点である。

そこで θ を 0 から π までを量子化した1次元の配列を4つ (ROH [], SUM [], MAXR [], MAXS []) 用意する。画像上のドット数を N として、 $(x-y)$ 座標上の任意の2点 $(x_1, y_1), (x_2, y_2)$ につき $(\rho-\theta)$ 空間での軌跡の交点、 (ρ_i, θ_i) を求め、 ρ_i の値を ROH [] に格納し、量子化した θ に近い SUM [] (θ_i) をインクリメントする。ある点 (x_i, y_i) に対して N 回 ($j=1, \dots, N$) 処理したあと SUM [] の極大点を MAXS [] に、その極大点の ρ_i の値を MAXR [] に各々格納する。

この処理を N 回 ($i=1, \dots, N$) 行い、最終的にもう一度 MAXS [] の極大点を拾い、これを求める (θ_0, ρ_0) とする。

MAXR [], MAXS [] に格納する際には前のラウンドの記録を更新することになるので、MAXS [] (θ_i) の値の大きい場合だけ書換えることにする。

このアルゴリズムでは全く同じ傾きをもち相対的な長さが極端に異なる線分を含む画像にたいしては有効ではない。何故なら MAXR [], MAXS [] は1次元であり、同一の θ に対して複数の ρ を格納できないからである。それゆえ、累積値の大きい ρ が残ることになる。

幸い、今回扱った画像は同じ傾きを持つ直線が複数存在するのだが、線分の長さがほぼ等しく、 θ を細かく量子化したことにより検出されない直線はなかった。

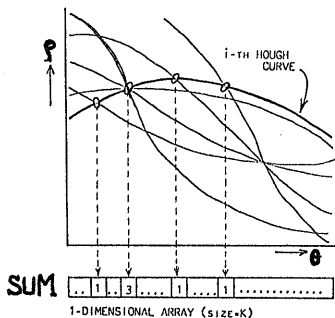


図9 1次元Hough変換

次に、各メッシュが表す直線の式に画像上の全点を当てはめて、誤差評価(6)をする。

評価式が真となる点を直線の構成要素とし、最終的に最小二乗法をこれらの構成要素について適用し、精度の高い傾きと切片を持った直線を検出する。

$$|y - (ax + b)| < \Delta \quad (6)$$

得られる傾き切片は以下の様な良く知られた式となる。

$$a = \frac{(n \sum xy - \sum x \sum y)}{(n \sum x^2 - (\sum x)^2)} \quad (7)$$

$$b = (\sum y - b \sum x) / n \quad (8)$$

この a, b を $y = ax + b$ に代入して回帰直線の式となる。

ここで、傾きの絶対値が1よりも大きいと実際には直線の近傍の点でも評価式の誤差が大きくなり、その直線の構成要素から外されてしまう点がでてくる。そこで、傾き絶対値が1よりも大きい $|a| > 1$ 直線に対しては

$$x = y/a - b/a \quad (9)$$

評価式 $|x - (y - b)/a| < \Delta$ として調べる。

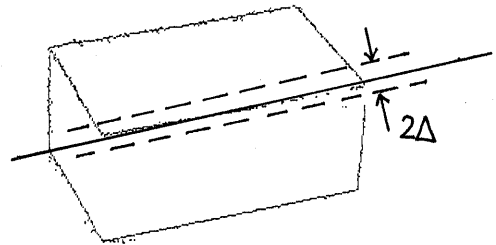


図10 最小二乗法適用範囲

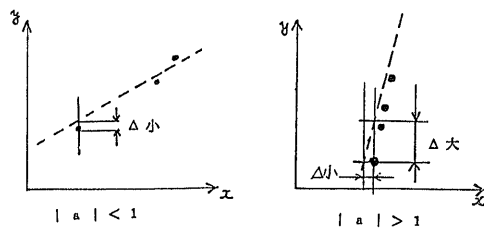


図11 誤差評価

3.2 稜線決定

前の段階で抽出された直線は傾きと切片だけなので、これでは不完全(図12)である。そこで頂点の検出をする。抽出された直線どうしの組み合わせから、全交点を求める。但し $0 \leq x \leq 510$, $0 \leq y \leq 479$ でない (x, y) については画面外ということでその交点は無視する。後でその交点がどの2直線の交点なのかを参照できるように、それらの情報を格納しておく。全ての交点の組み合わせによって得られた稜線候補群を図13に示す。画像の稜線は稜線候補群の中に必ず存在する。

次に求められた交点から頂点候補を選抜する。まず画像を表示し、各交点中心に $5 * 5$ ドットの枠内に画像上に点が存在するかどうかを調べ、1つも点が存在しなければ、その交点を頂点候補から外す。前処理においてかなり面密に雑音除去を施してあるので、この方法により画像のエッジ上か、かなり近傍の点以外は頂点候補として認められない。

頂点の選抜の結果は、現在与えられている直線が正しく再現できるものでなくてはならない。そこで今度は、与えられた直線を探索対象とする。

与えられた直線 N 本(各直線を $L_i, L_j; 1 \leq i, j \leq N$ で表す)、残った頂点候補 M 個(各頂点を $P_k; 1 \leq k \leq M$ で表す)、交点としてのどの2直線による交点なのかという属性の情報が各々2ずつ(頂点 P_k の属性を $I(P_k, 1), I(P_k, 2); I(P_k, 1) = L_i, I(P_k, 2) = L_j$)。

全ての $I(P_k, 1) = L_i$ 、または $I(P_k, 2) = L_i$ となるような P_k は L_i に属すると考える。尚、少なくとも、 L_i に属する頂点は2個以上は必ず存在する。

各 L_i に属する全ての P 間で距離が最大ものを L_i から薄びかれる稜線と仮定する。 L_i に属するのが P_k, P_l と2つだけの時には、 P_k, P_l を結んだものを L_i から薄びかれる稜線と決定する。(こうすることで2次元画像の輪郭線は Hough 変換でしきい値をパスしている限り、かなりエッジが途切れていても再現が可能である。)

頂点を3個以上もっている L_i については、必ずしも最大のものが真の稜線とはならない(図14)。 L_i 上に存在する画像上の点の合計を S とする。 L_i に属する全ての頂点間で線分 L_{ij} (または $P_k P_l, j; L_i$ に属する頂点数) を引き、 $S / |L_{ij}|$ が最大であるものを稜線と決定する(図15)。

図形は閉じているので、近傍の頂点は平均(複数の点の重心)を求めることにより1つの頂点と見なすこととする。こうして稜線表を作成し、得られた図形が図16である。

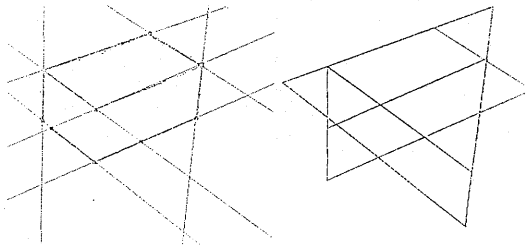


図12

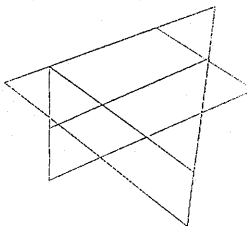


図13

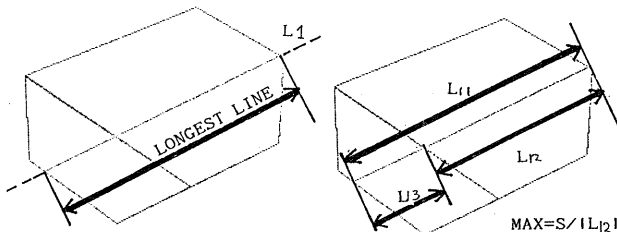


図14

図15

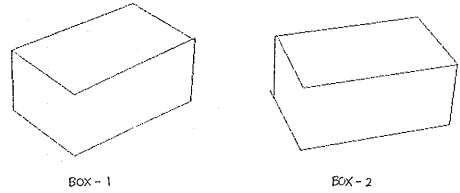


図16 稜線図形

3.3 図形補充

図16を見ると頂点の状況、線のつながり、稜線の欠損など人間の目からみて明らかに不自然な図形となっている。そこで、(1)頂点の修正、(2)稜線の補充、という順序で図形の補充をおこなう。予め各頂点のつながりを示す頂点リストを作成しておく。

(1) 頂点の修正

1. 図形の輪郭線を探査し、記録する。
2. 輪郭線上の頂点を検出する。(図17)
3. 稜線図形より輪郭線以外の稜線を削除する。(図18)
4. もう1度輪郭線探索をして、消えた輪郭線上の頂点があれば、その近傍の頂点が本来の頂点と見なす。
5. 頂点リストと輪郭線上の頂点より本来存在しない稜線を検出する。(図19)
6. 図18の頂点AかBのどちらが矛盾するのかを調べる。AをBに、またはBをAに重ね、輪郭線の一筆書きのできなかった場合、重ねられた側の頂点が矛盾する点とみなし、重ねた側の頂点に置換える。

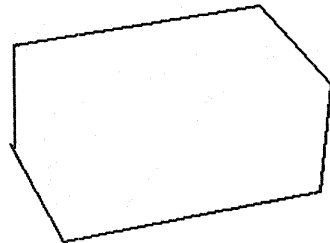


図17

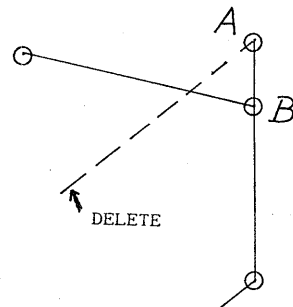


図18

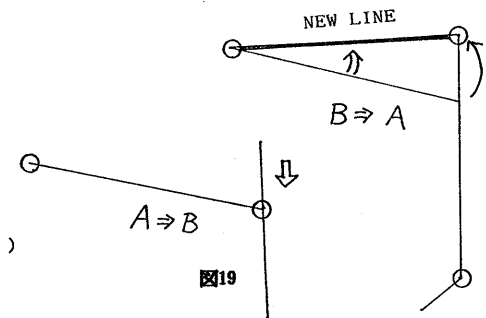


図19

単一の多面体では Huffman の線画パターンの T 型は含まないとすると、常に頂点と頂点は結びつき、稜線上の端点や孤立した頂点は存在し得ない。つまり A が B のどちらかがおかしいということになり、これはもともと A と B が同一の頂点であったといえる。どちらかを一方に頂点座標を重ねて一筆書きをすると矛盾する点を重ねたとすると新しい稜線がディスプレイ上の図形に現われる。

画像を修正する場合、少なくとも輪郭線はあい味ではないということをも前提にしている。実際には Hough 変換後の誤差評価式の条件を緩めて構成要素をふやせば、頂点の矛盾は起りにくいと思われる。

(2) 図形補完

1. (1) 処理後の図形に対して Huffman の線画パターンによる稜線の符号化を行う。輪郭線を描成する稜線へで、図形内部は凸稜線として、+ の符号を与える。これは最初に扱う対象を凸多面体に限定しているためである。
2. 地面に接している稜線に対し、凹であるマイナスの符号をつける。
3. 各頂点において、Huffman の線画パターンを満たさない頂点を抽出する。
4. 矛盾する頂点どうしを結び、新たにこの稜線を頂点リストに付け加える。

Huffman の線画パターンは実在する図形に対しては条件を満たすが、これを満たすからといって必ずしも与えられた図形が実在するとは限らない。今回の図形も同様であり、何等矛盾は生じないが、この図形は不自然である。よって、この必要条件の他に拘束条件として、多面体が空中に浮いているのではなく、地面に接しているという状況を与えてやる。

すると、B は矛盾を呈し、A は矛盾を生じないので、初めからの矛盾点 C と B を凸であるプラスの符号をつける。接地している稜線は垂直に近い稜線とつながる Y 座標の大きい側の頂点間の稜線を描出する。

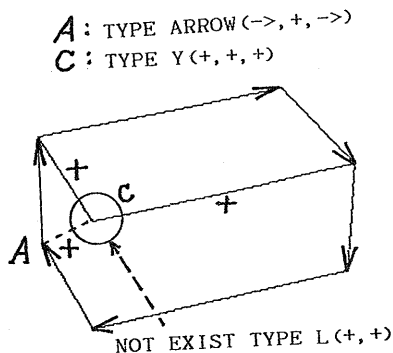


図20 Huffmanの線画パターン適用図

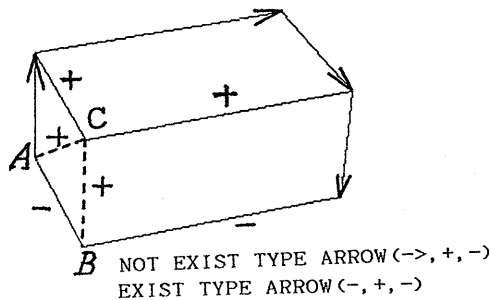


図21 必要条件のみの場合と拘束条件を加えた場合

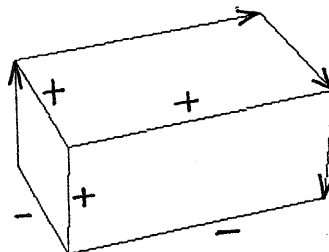


図22 補完された図形 (BOX 1 と BOX 2)

4 3次元再構成

4.1 頂点の対応付け

ワールド座標を決定するためには、視点の異なる2枚の画像を使用するが、その2枚の画像間の頂点の対応付けが必要である。(図22)

カメラ視点から画像1の頂点へ向かう直線はその直線のワールド座標を通り、その直線を別の視点から写した画像2上に投影したものが Epipolar line であり、そのline上に前の画像1上の頂点に対応する頂点が存在すると考えられる。

そこで Epipolar line と画像2上のすべての頂点の距離を計り、許容範囲内に存在する頂点が1つの場合はそれに対応付ける。複数の場合は逆に画像2上のそれらの各頂点から画像1へ Epipolar line を作り、対応付けを行っている画像1の頂点との距離が最小の Epipolar line が画像2での同一のものとして対応付ける。

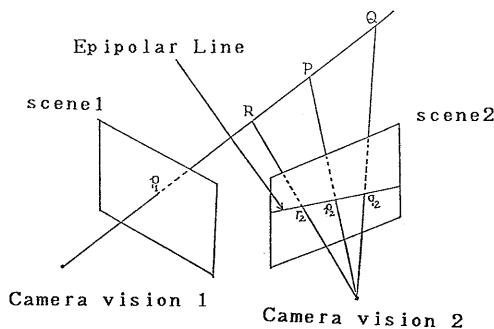


図22 Epipolar line

4.2 カメラ・パラメータ

ここでカメラ・パラメータについて考察してみる。まずカメラは次の様に設定する。

(1) ターン・テーブルの回転中心をワールド座標の原点とする。

(2) カメラの光軸をその原点に一致させる。

カメラの光軸がZ軸上にくるように、Y軸回りに β 、X軸回りに α だけ回転し、さらに光軸がフィルム面上で水平に成るようにZ軸回りに γ だけ回転した時の変換マトリクス T_1 は次のようになる。

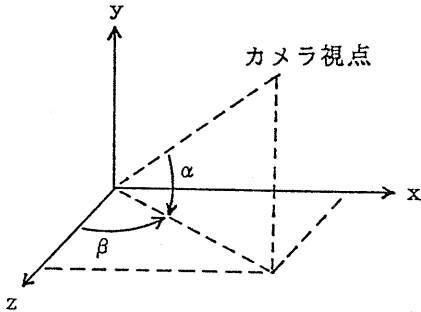


図23 ワールド座標系

$$T_1 = \begin{bmatrix} \cos \beta & 0 & -\sin \beta & 0 \\ 0 & 1 & 0 & 0 \\ \cos \beta & 0 & -\sin \beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \alpha & \sin \alpha & 0 \\ 0 & -\sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} \cos \gamma & \sin \gamma & 0 & 0 \\ -\sin \gamma & \cos \gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1/\alpha \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad T_3 = \begin{bmatrix} k & 0 & 0 & 0 \\ 0 & k & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

ここでカメラ視点と原点の距離で、カメラ視点とスクリーン平面の距離を1として、 $k=1/d$ である。これより、CRT座標への変換マトリクスは次のようになる。

$$T = T_1 T_2 T_3 = \begin{bmatrix} T_{11} & T_{12} & 0 & T_{14} \\ T_{21} & T_{22} & 0 & T_{24} \\ T_{31} & T_{32} & 0 & T_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

ここで、

$$\begin{aligned} T_{11} &= k (\cos \beta \cos \gamma - \sin \alpha \sin \beta \sin \gamma) \\ T_{21} &= -k \cos \alpha \sin \gamma \\ T_{31} &= k (\sin \beta \cos \gamma - \sin \alpha \cos \beta \sin \gamma) \\ T_{12} &= k (\cos \beta \sin \gamma - \sin \alpha \sin \beta \cos \gamma) \\ T_{22} &= k \cos \alpha \cos \gamma \\ T_{32} &= k (\sin \beta \sin \gamma - \sin \alpha \cos \beta \cos \gamma) \\ T_{14} &= \cos \alpha \sin \beta / d \\ T_{24} &= -\sin \alpha / d \\ T_{34} &= -\cos \alpha \cos \beta / d \end{aligned}$$

これより、ワールド座標 (x, y, z) とCRT座標

(x^*, y^*) との関係は

$$\begin{bmatrix} x & y & z & 1 \end{bmatrix} T = h \begin{bmatrix} x^* & y^* & 0 & 1 \end{bmatrix}$$

$$T_{11}x + T_{21}y + T_{31}z = hx^*$$

$$T_{12}x + T_{22}y + T_{32}z = hy^*$$

$$T_{14}x + T_{24}y + T_{34}z = h - 1$$

となり、 h を消去して次の式が得られる。

$$\begin{aligned} (T_{11} - T_{14}x^*)x + (T_{21} - T_{24}x^*)y + (T_{31} - T_{34}x^*)z &= x^* \\ (T_{12} - T_{14}y^*)x + (T_{22} - T_{24}y^*)y + (T_{32} - T_{34}y^*)z &= y^* \end{aligned}$$

この式は5点の座標 (x, y, z) と (x^*, y^*) が与えられなければ、9個のカメラ・パラメータが求まることになる。尚、一方のカメラ・パラメータが求まれば、他方は y 軸回りの回転角度が異なるだけであるから簡単に求めることができる。

結局、カメラ・パラメータが求まれば、2つの透視投影 T^1, T^2 による画像上の点 (x^{*1}, y^{*1}) と (x^{*2}, y^{*2}) から対象の点の座標 (x, y, z) の近似値は次式で表される。

$$X = (A^T A)^{-1} (A^T B)$$

$$X = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad B = \begin{bmatrix} x^{*1} & y^{*1} & x^{*2} & y^{*2} \end{bmatrix}^T$$

$$A = \begin{bmatrix} T_{11} - T_{14}x^{*1} & T_{21} - T_{24}x^{*1} & T_{31} - T_{34}x^{*1} \\ T_{12} - T_{14}y^{*1} & T_{22} - T_{24}y^{*1} & T_{32} - T_{34}y^{*1} \\ T_{11} - T_{14}x^{*2} & T_{21} - T_{24}x^{*2} & T_{31} - T_{34}x^{*2} \\ T_{12} - T_{14}y^{*2} & T_{22} - T_{24}y^{*2} & T_{32} - T_{34}y^{*2} \end{bmatrix}$$

5 考察

各処理の所要時間を表に示すとともに、評価する。

- (1) 全てパソコン上の簡易な処理システムとなった。
- (2) 2次元配列のHough変換と比較して1次元配列のHough変換では、所要メモリがかなり節約できたが、計算量が二乗のオーダー違うので、所要時間も二乗となっている。
- (3) Hough変換によるエッジの当はめは高精度であり、分解能も高く、エッジに欠損があっても検出が可能である。
- (4) 相対的に短いエッジが存在すると、直線として抽出されない恐れがあり、そのような状況に遭遇すれば、不自然な2次元画像が再生されることになる。幸い、Huffmanによるパターンマッチングにより図形の補完はできるが、1次元配列使用の際、特に平行な線分に対する対策が必要である。

(5) 対応付けはエッジ上の点と点で行うのが一般的だが今回は頂点間での対応付けを用いたので、処理の高速化が実現できた。

(6) カメラ・パラメータは誤差に敏感なので、行列式による処理以外の方法を考察するのが好ましいと思われる。

(7) 対象を彩色したり、色照明などにより、エッジの接続状態やカラー情報を取り込み、より安定した輪線の抽出が今後の課題である。

今回は全ての処理をパソコン上で実現しているので、コンパクトなシステムとなっている。

一連の処理の中にはしきい値や精度などの設定が対話形式になっているので、濃度やメッシュのヒストグラムから適当な値を決定できるよう画像処理の自動化の実現も今後の課題として望まれる。

参考文献

- [1] 山口富士夫 : 「コンピュータ・グラフィックス」
(日刊工業新聞)
- [2] R. Nevatia: 「画像認識と画像理解」(啓学出版)
- [3] 安居院益、中島正之: 「コンピュータ画像処理」
(産報出版)
- [4] 尾崎弘、谷口慶治、小川秀夫:
「画像処理—その基礎から応用まで」
(共立出版)
- [5] 長尾真 : 「デジタル画像処理」(近代科学社)
- [6] 長尾真 : 「パターン認識と図形処理」(岩波書店)
- [7] 松山隆司、長尾真 : 「Hough変換の幾何学的性質と
直線群検出への応用」
情報処理学会論文誌 Vol. 26 No. 6
- [8] 奥水大和: 「直線群検出のためのHough曲線追跡型アル
ゴリズム」
電子通信学会論文誌 Vol. J69-D No. 4