２値化画像からの図形形状パラメータ抽出の高速化に関する研究

平成９年６月

中　島　勝　行

2値化画像からの図形形状パラメータ抽出の高速化に関する研究

中島勝行

第１章　緒論

１．１　緒言

画像信号は音声信号あるいは工業計測の際に現れる信号に比し単位時間当たりの情報密度がきわめて大きい．この信号をディジタル計算機を用いて平均化，微分などのフィルタリング，フーリエ変換等の処理を行い，その画像特徴パラメータを抽出する作業には多くの時間を要する．ある種の実験結果から得られた写真，あるいはディジタルイメージをもとに解析を行うような場合，時間的制約は比較的少なくソフトウエアによる処理で問題が解決することも多い．時間的制約が多少厳しくなるとＤＳＰ（ディジタルシグナルプロセッサ）などの専用ハードウエアが用いられるようになる．ＤＳＰではデータバス，アドレスバスの複数化，メモリの分割並列アクセス等の工夫がなされているが処理の手順はＲＯＭ化されたプログラムとして記述されており，ノイマン型計算機の基本的な考え方からは脱却していない．それ故，自ずと処理速度の限界が存在する．実時間処理（どの程度が実時間であるかは問題であるが）に対しては，処理速度に対する要求はさらに厳しくなり，画像に対する処理の特定の内容に対応した，専用の論理回路（専用ハードウエア）を構成し処理を実行することになる．専用ハードウエアを構成する場合，ＤＳＰを用いる場合もそうであるが，処理手順がパラレルに実行できるというメリットを最大限に活かして，適用するアルゴリズムを選定する，あるいは新しく作成するということが非常に重要になってくる．さらにその論理構成が，可能な限り簡潔であることも，回路作成に要するコストの点で重要な問題である．本論文は以上に述べた画像の，ソフトウエア，ハードウエア処理に関して，主に処理速度の向上，さらに計算機リソースの有効利用等の観点から考察を行っている

１．２　本論文の背景と研究の位置付け

本論文の研究内容は２値化図形形状パラメータ抽出手法分野の中で

(a)画素形状とディジタル測長，面積、形状係数等の測定に関する諸問題についての考察（第２章）．

(b)(a)を応用した輪郭長，面積，周囲長，形状係数の計測に関するもの（３～５章）．

(c)Hough変換アルゴリズム実行の高速化，高効率化に関するもの（６～１０章）．

(d)以上の理論を統合的に，実用に供する実施例（１１章）．

について述べている．１２章では以上の研究成果を総括して述べている．なおここで言う”**ディジタル測長**”とはディジタル図形の濃度等高線に沿っての２点間の長さと定義する．次に(a)～(d)についてこの研究分野の背景とその位置付けを示す．

１．２．１画素形状とディジタル測長，面積、形状係数等の測定に関する諸問題について

（１）縦長画素の実現．

画像を構成する画素の形状としては従来からあるものとして正方形画素，６角形画素によるものが挙げられる．前者については画像の輪郭を記述する上で４連結で取り扱うか８連結で取り扱うか、いくつかの問題を含んでいる．これに対して６角形画素を用いる方法は画像平面上で異方性を持たない画素の記述法として理論的取り扱い上好んで用いられる．

しかし実用上の撮像装置としてはラスタースキャンニング方式が大部分で,この場合撮像時点での画素構成は必然的に正方形或いは長方形にならざるを得ない．画素を正方形にするか長方形にするかについては映像信号のＡ／Ｄ変換のサンプリング周期と密接に関連しており，ディジタル回路による処理を前提とすれば（計算機でのソフトウエア処理を含む．）サンプリング周期を短くすれば縦長画素として捉えられ長くすればその逆である．また現在のＮＴＳＣ方式の画像伝送では飛び越し走査が行われており1/60[s]の周期で奇数、又は偶数フィールドの走査が終わり1/30[s]で１画面分の伝送が終わる．もしここで奇、偶いずれかのフィールドのみをサンプリングすれば２倍の縦横比の画素として捉えられる．このように縦長画素を実現する手段は２つある．

（２）縦長画素とディジタル測長との関連

従来からあるディジタル測長法には様々なものが考えられている．これは１．２．２項で取り上げるが本論文では第３章で”隣接２線走査法（ＤＬＳ法）”，第４章で”スキップ２線走査法（ＳＤＬＳ法）”を提案している．この方法によると画素を縦長に採ることにより測長精度が大幅に向上するというものである．そしてその画素の縦横比には最適値が存在することを明らかにしている．前者のＤＬＳ法は専用ハードウエアを用意する方法で後者のＳＤＬＳ法はソフトウエアにおける手順である．但し後者もハードウエア化することは可能である．

（３）画像入力との並行処理

従来からある図形形状パラメータ検出に関する理論は一旦計算機のメモリに取り込んだ後,それらの理論を適用するものがほとんどである．画像入力と並行してして処理を行おうとするとそのアルゴリズムをハードウエア上に実現することは困難な場合が多い．本論文で提唱する手順はいずれも簡素なものでハードウエア化が容易であって，１～２走査線分の、メモリを用いて画像入力との並行処理を実現している．また実際にＩＣ，ＬＳＩを用いてそのアルゴリズムをハードウエア上にインプリメントし数多くの実験を行い，その高速性，精度の向上を確認している．

（４）与えられた図形の２値化の過程について

本論文で取り扱う対象は２値化画像データであることを前提としているから，撮像装置により捉えられた画像はある閾値を持って”1”または”0”のデータに変換される．一般の画像処理に関する文献では，図形が当該画素の中心を覆っているか否かでその閾値を定めている例が多い．しかし本論文で取り扱う輪郭長等の測定の際には，この２値化による階段状の凹凸が，測定精度に大きく影響を与えることが判明している．また映像信号上に重畳される微細なノイズが２値化パターンの形状を大きく左右する．そこで本論文では撮像装置の，光を検知する一画素分のセンサの表面を，図形がどのように覆っているかについて，その閾値を面積比率で求め”1”,”0”を決定することにしている．対象としては矩形図形と円弧を採り上げた．

１．２．２　面積，輪郭長，周囲長，形状係数の測定について

対象図形の２値化されたイメージが得られているとき，面積の計測はごく簡単である．つまり画素値が”１”である画素の個数を数える，一方向からの一連の走査で完結する．測定の精度は測定系の面分解能に依存し，測定アルゴリズムに大きな改良の余地はない．但し視野内に複数個の図形が存在し，個々にその面積を求める場合は多少の工夫を要し各図

形F_1，F_2，-- F_nに対する連結性を考慮しながら対応する面積カウンタS_1，S_2 -- S_nを＋１する作業を行わなければならない．しかしこれはそれほど困難な作業ではない．

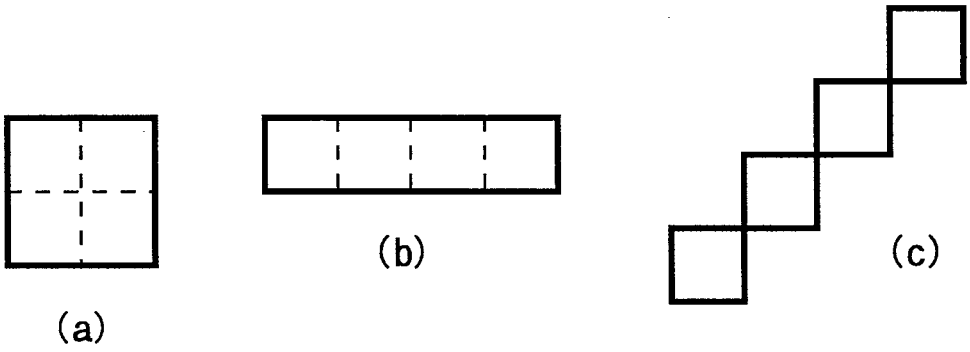


Fig.1.1 3 Kinds of pattern

Table 1.1 Mearsured perimeters for each pattern

	(a)	(b)	(c)
(1)	8	10	16
(2)	4	6	6
(3)	4	6	$6\sqrt{2}$
(4)	4	4	4

次に輪郭長，周囲長，形状係数の測定について考察してみる．図形の周囲長測定の古典的手法は Rosenfeld & Kak による"ディジタル画像処理"[2]に示されている．これにはいくつかの手法が紹介されているがこれを列挙すると次のようになる．

(1)図形Fとその外側$\overline{F}$を分離する境界の長さの和として求める方法
(2)図形Fの境界追跡アルゴリズムでFの境界を追跡するに要するステップ数として求める．
(3)(2)と同様であるが８連結の斜め方向に対して$\sqrt{2}$を割り当てる方法．
(4)図形Fの境界点の数，すなわちFの空間微分F'の画素"１"の点の数を数える方法．

図１．１に示す３種類の２値化パターンに対してそれぞれの手法を適用して周囲長を求めたものが表１．１である．(1)～(4)の方法で求めた値には大きな差異が認められる．詳細は第２章で述べるが，２値化される以前の原図形との対応関係を十分検討していずれの方法が測定精度が高いかを調査する必要があることをうかがわせている．

周囲長測定に関しての他の研究として" ３画素ベクトル法"[3-5]がある．この方法では三つの連続した画素を基本単位として，各単位自体の長さと，相隣る基本単位の連結状態とを考慮して，長さを求めるものである．この手法は金属組織などの，粒子解析を目的

として開発されたもので，比較的小さい図形サイズまで高い精度で測定が可能である．しかし本論文の主たるテーマである，高速実時間処理を行うためには，測定手順のハードウエア化が必須であるが測定アルゴリズムが複雑なため，困難な点が多い．

そのほか１．２．３項で述べるHough変換の拡張として周囲長を求める方法も提案されている．下図は以上の研究成果を整理したものである．

以上ディジタル図形の周囲長測定という観点から従来手法について述べた．周囲長は図形輪郭を一周した，閉じた輪郭長である．またここで言う輪郭長は２値化されたディジタル図形の２点間の長さの測定と考えてよい．何らかの効率的且つ高精度なディジタル測長法が考案されれば，それは直ちに周囲長測定に適用可能である．

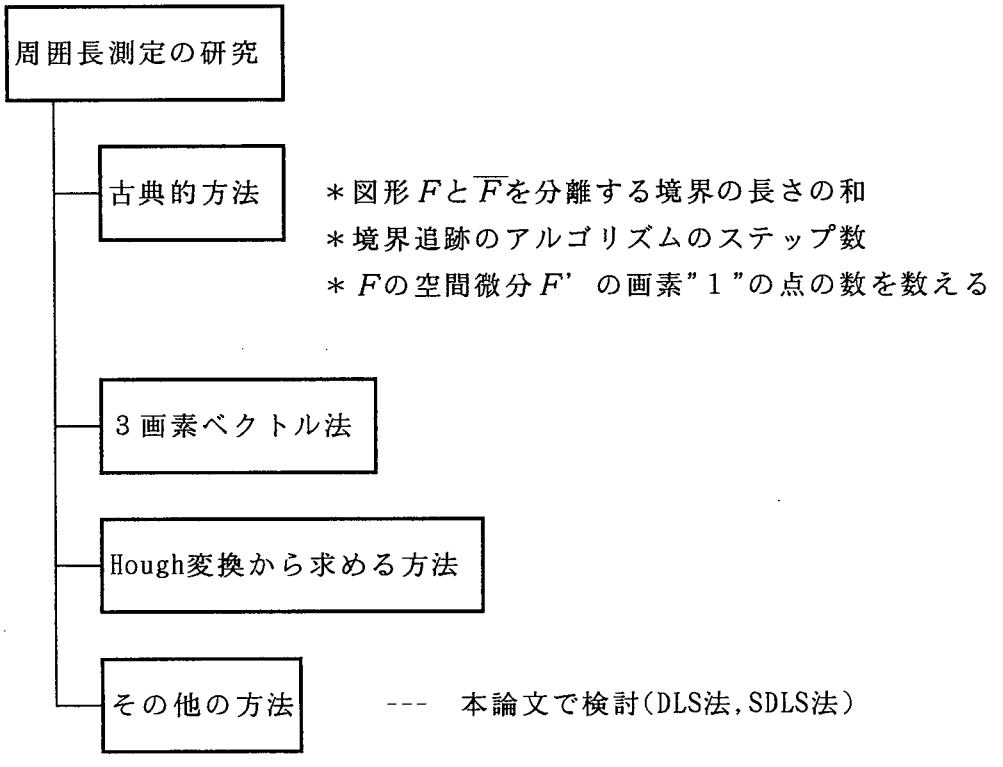


Fig.1.2 Tree of research about perimeter measurering

周囲長測定に関連した図形形状パラメータとして形状係数がある．形状係数の定義はいくつか考えられるが本論文では形状係数Fを次式で定義して用いる．

$$F = L^2 / 4\pi S \qquad \text{---(1.1)}$$

但しLは周囲長，

Sは面積である．

１個の図形を想定した場合，この定義によれば凹凸が最も少ない図形，すなわち円のときFの値は１となり多角形，四角形，三角形となるに従って１より大きくなっていく．形状係数は１個の粒子を想定した場合，金属などの粒子形状の凹凸の度合いを表す．また視野内に多数の粒子が存在する場合，これを全体として捉え，その値を計測すれば粒子の細かさを表す指標となる．

１．２．３ Hough変換アルゴリズム実行の高速化，高効率化

画像上の直線を検出する代表的な手法としてHough変換がある．これはP.V.C.Houghが1962年米国の特許として，傾き-切片座標系において直線検出する手法を提案したのが最初で，Duda & Hartはこれを$\theta-\rho$ヒストグラム平面に置き換えて直線を検出する方法を示した．この変換式は式(1,2)で示される．

$$\rho = x \cdot \cos\theta + y \cdot \sin\theta \qquad \text{---(1.2)}$$

この変換の様子を図1.3に示す．X-Y平面上の点A,B,C,D 及び Pは$\theta-\rho$ヒストグラム平面上の正弦曲線A,B,C,D 及びPにそれぞれ対応する．このようにX-Y平面の１点に対して１つの正弦曲線が対応するので計算量が非常に大きくなる．しかし Hough変換はその耐ノイズ性，耐隠蔽性といった意味で優れたパターン検出手法として注目され，その後この手法に関する各種の研究が活発に行われている．その主な方向は

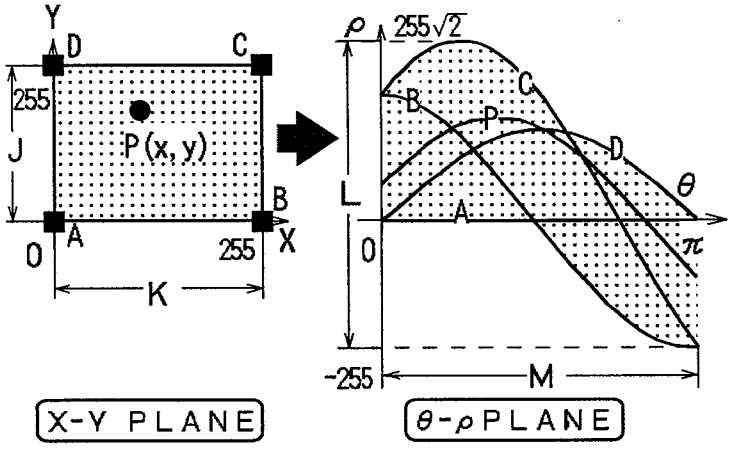


Fig. 1.3 Relation between X－Y plane and θ － ρ plane

(1)円、楕円，その他任意形状パターン検出への一般化[29],[30]，
(2)ヒストグラム平面離散化問題とHough変換精度との関係，
(3)この手法の最大唯一の欠点である計算コストの大きさを低減する試み，[21],[23]－[28]

である．文献(11),(30)によればHough変換によって原図形中の直線の位置，周囲長，フェレ長，絶対最大長，面積，形状係数などが算出可能なことが示されている．また任意の図形と相似な図形を抽出することも可能である．特に(3)に関しては実用化に際し最も障害になる問題として数多くの研究がある．巻末に挙げた文献はこれらに関するものがほとんどで，本論文の目的とするところもここにある．この計算コストの問題は３つに分けられる．それは

①$\theta-\rho$ヒストグラム平面のρアドレスを特定するためのρ値計算に要する時間，或いはメモリサイズの問題
②$\theta-\rho$ヒストグラム平面の特定の要素のインクリメント操作に必要な時間の問題
③$\theta-\rho$ヒストグラム平面の累積度数の探索

である．①ρ値計算の時間の問題に関しては解決方法として

(a)Hough曲線をアナログ回路で発生させAD変換を経てメモリに入力する方法[19]
(b)DSPなどのハードウエアに近い演算素子を用いる方法[22]
(c)対象図形が持つある特徴に注目して変換点の数を減少させる方法[21],[26],[31]-[33]
(d)Hough変換式中の正弦関数を表で用意する方法[25],[34]
(e)連立漸化式を用いてρ値をインクリメンタルに発生させる方法[23],[24]
がある.

(a)は振幅と位相が制御可能な発振器を正弦関数計算に用いてHough曲線を発生させこれをA/D変換してメモリに入力する方法である.アナログ信号を用いるので精度的には限界があると思われる.

(b)DSPを用いた方式は文献(22)で実用に供した報告がなされている.しかしハードウエアは相当に複雑でコストも大である.

(c)については自動車のナンバープレートの検出などに実用レベルでの研究が行われているが,適用対象に制限が加えられ原方式の汎用性を低下させているものも少なくない.

(d)の方式は文献(25)で大型計算機上で表の形式の変更(配列の次数)による速度比較が行われているが表のサイズを大きくとっても速度の向上はあまり期待できないという否定的な結果が出ている.この主な原因はOSのオーバーヘッドタイムと,参照しようとする配列の実アドレス計算に整数の乗算と加算が含まれるからである.アセンブラでHough変換の処理を記述するとこの整数乗算,加算の影響が大きいことが明確になる.たとえば3次元の表を配列として用意するとして,その要素を参照しようとすると2回の整数乗算と2回の整数加算が必要である.

大型計算機が専用の浮動小数点演算装置を持ち,整数乗除算或いは加減算との演算時間の差が少なくなるほどこの問題は顕著になる.この種の問題についてその解決方法を第7章で検討する.

(e)は正弦関数を連立漸化式を用いて発生させるものでコンピュータグラフィックスにおける円の高速描画に用いられる手法にその根拠を置いている.この方法は上記メインメモリの参照を必要とせず,CPUのレジスタ間のシフトと加減算のみでρ値の発生ができ,速度の点では画期的に速くなっている.また作業用のメモリを必要としないことも注目すべき点である.この方法について第8章で議論する.

次に②θ-ρヒストグラム平面の特定の要素の度数累積操作に必要な時間を短縮する問題について述べる.

この手順は特定の要素の内容の読み出し,+1操作,再書き込みの3段階に分かれ,ハードウエア,ソフトウエアいずれで実行しようとも手順縮小の余地は少ないと考えられるが,発想の転換を図り,光学系とCCDイメージセンサを利用してこれを高速に実行する方法を9章で提案している.

③については主にソフトウエア上で実行することが多く,且つ応用の用途によってもバリエーションがあるので本論文では深い考察は行っていないが第8,9章では提案した原理の副次的効果として度数探索の範囲を限定できるようになり,高速化が期待できることを述べている. 図1.4はHough変換に関しての研究分化の様子を本論文のテーマを中心に整理して示したものである

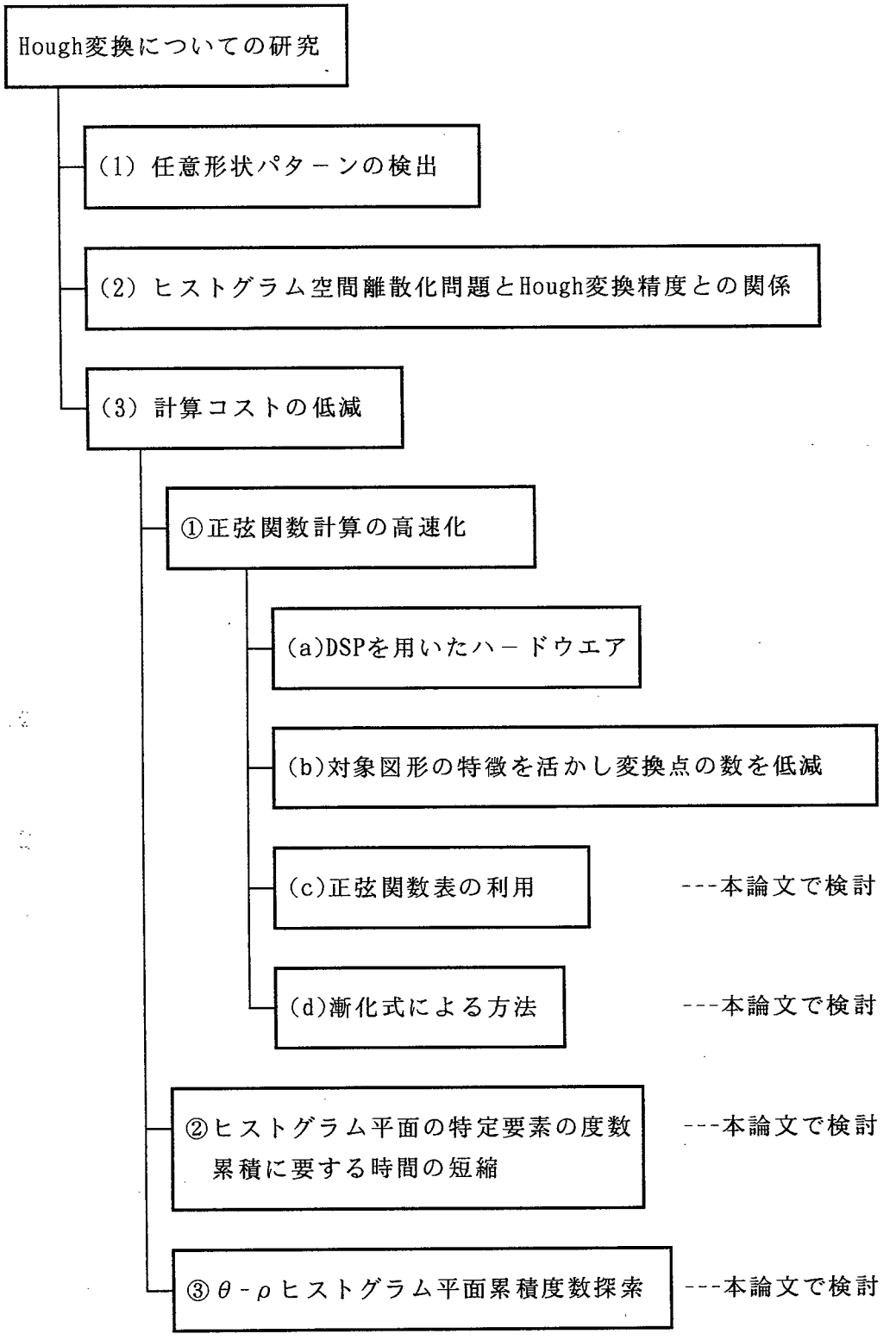


Fig.1.4　Tree of research about Hough transform

1．2．4　矩形から構成される図形の検査

この分野の研究は特に矩形に限定することなく，様々な手法が開発され，すでに多くの

実用化された装置が稼働している．本論文では高速化された，形状係数の測定及び，Hough変換手法を融合させ，主に直線成分で構成される図形の欠陥を検出する実用装置に関しての考察を行っている．具体的には養殖海苔の乾燥検査に上記手法を応用した例を第１１章で取り上げている．

１．３　本研究の目的

本研究は個々の内容的には１．２項で述べたように２値化されたイメージから形状パラメータを高速に抽出し，実時間での図形の形状分類へ応用することである．その過程として図形のディジタル測長，２値化の問題，形状係数測定の際の高速化アルゴリズムの開発，Hough変換の高速化の問題を取り扱う．図形の２値化については直線エッジ，及び円弧で構成されるエッジがどのような条件のもとに"1"/"0"に切り分けられるかを検討する．また元映像信号上に乗るノイズが２値化パターンにどのように影響を与えるかを検討し周囲長，面積の測定値の変化を調べる．

ディジタル測長に関しては専用ハードウエアを用いた実時間処理システムについて考察し，イメージ信号レベルでの高速処理をめざす．また実時間処理用に開発した処理アルゴリズムをソフトウエア処理に適用できるように拡張しその性能評価を行う．

Hough変換に関しては正弦関数を高速に発生させる方法としての表参照方式を解析し，より効率的な表の利用，少ない容量の表による参照をめざす．また連立漸化式による方法を用いてより高速、且つ高精度な計算式の開発をめざす．またθ-ρヒストグラム平面の度数累積方法について，新しい概念としてCCDイメージセンサ上に度数累積を行う方法を検討する．これはイメージセンサが個々の光感知素子に並列に接続されたコンデンサに光起電力を積分蓄積する現象を利用するものである．以上の手法を効果的に融合させ，輪郭が直線からなる図形（矩形）について実時間で，その欠陥を抽出するシステムの開発を行う．

１．４　本論文の構成

図１．５に本論文の構成を示す．本論文ではまず第１章で本論文の背景と研究の位置付けを各分野ごとに示す．ついで第２章では第３．４章に示す新しいディジタル測長法の基本的な考え方としての縦長画素採用の効果の定性的な考察を行う．また画像入力と並行して処理を進める高速性の追求について言及する．また直線及び円弧について，これが２値化パターンに変換される過程を考察する．また分解能，重畳ノイズとの関連を検討する．第３章では専用ハードウエアを用いたビデオレートでの輪郭長測定の原理（隣接２線走査法---ＤＬＳ法）を示し，画素を縦長に採った際の輪郭長測定精度の向上の様子を直線エッジについて定量的に示し，円弧エッジについてはシミュレーションにより，その結果を示す．さらにこの原理にもとづき，専用ハードウエアを作成し，そこでの各種図形の周囲長，形状係数測定結果を示す．特に図形の姿勢，図形サイズが小さくなったときの測定精度について詳細を示す．第４章では３章で開発した隣接２線走査法をソフトウエア処理にも応用出来るよう拡張したスキップ２線走査法（ＳＤＬＳ法）についてその原理を述べる．ついで同じくシミュレーションによる四角形，円弧についての測長結果を示す．第５章で

は農産物製品としての甘藷及び建築材料であるコロニアル瓦を対象にディジタル測長の応用実施例を示す．第６章ではHough変換の実行過程を分析し後章で述べる新しい処理方法の基本的な考え方を示す．第７章では表参照方式Hough変換に於ける表の高速読み出しと，表サイズの削減を兼ねた極座標型表参照方式高速Hough変換と分割縮小化表参照方式高速Hough変換について考察する．第８章では連立漸化式を用いたρ値計算について考察し，座標回転行列から導かれる各種の高速Hough変換用漸化式について述べ、これがHough変換用各種漸化式の基本となっていることを示す．第９章では電荷蓄積素子による度数累積の高速化について議論する．また累積度数の探索の高速化の補助回路について述べる．第１０章ではHough変換の応用例を示す．第１１章では以上の手法を矩形図形の欠陥抽出に応用して，養殖海苔の不良品摘出検査への実際的な応用例を示す．第１２章では本研究にて得られた成果を総括して述べる．

本論文の構成フロー

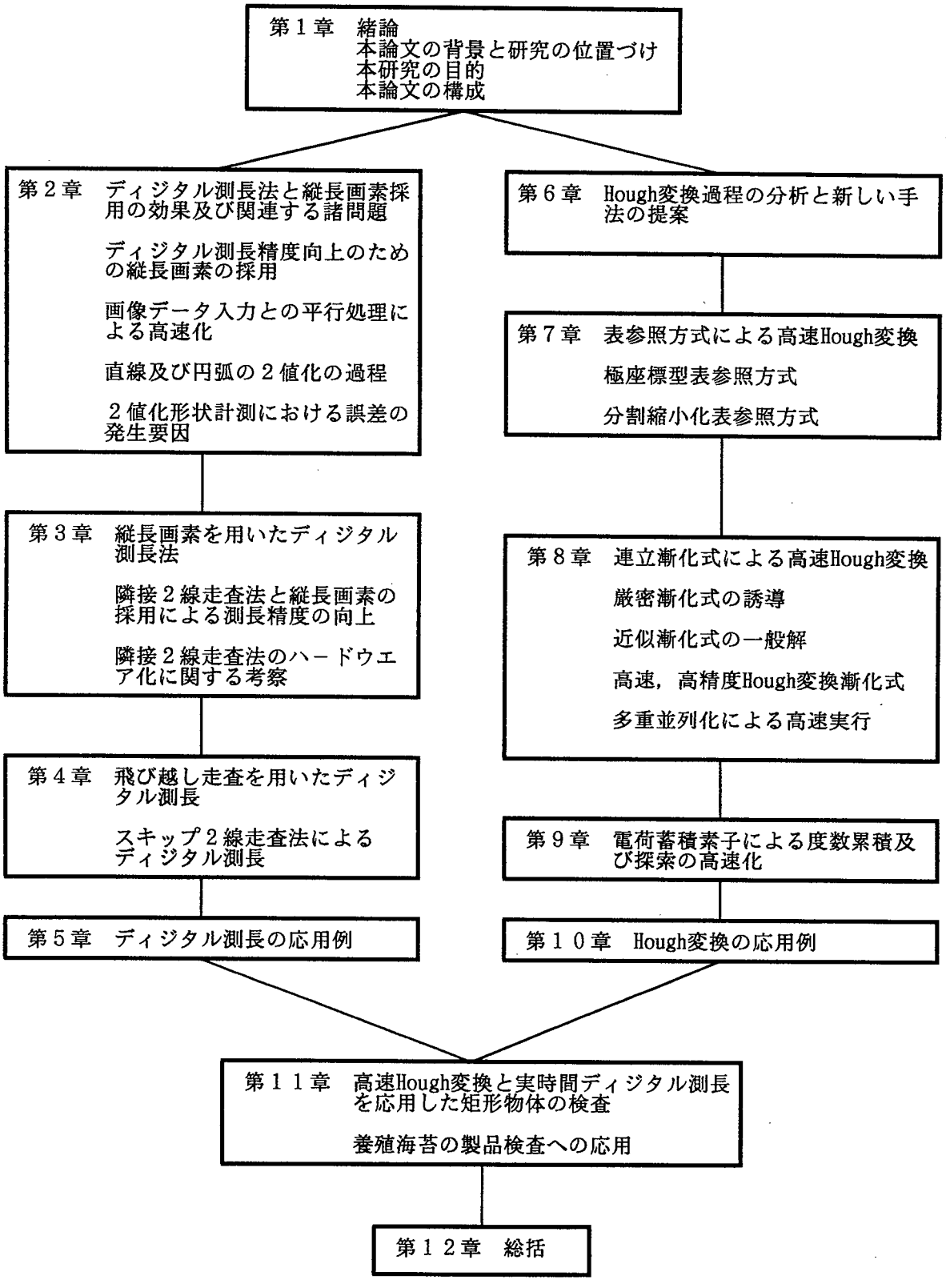


Fig.1.5 Flow chart of this study

第２章　ディジタル測長法と縦長画素採用の効果及び関連する諸問題

第１章においてディジタル測長の概念とこの分野の研究の背景を述べた．この章では本研究の主たる成果の一つである隣接２線走査法（ＤＬＳ法）及びスキップ２線走査法（ＳＤＬＳ法）について，両者に共通して取り入れられている縦長画素採用による精度及び処理速度向上についての基本的な思想を定性的な面から述べる．具体的なアルゴリズムとその性能評価については後続の第３章，第４章にてそれぞれ詳細に述べる．またディジタル測長法の精度を定量的に評価するには原図形と２値化されたときの図形との関係が重要である．このため直線エッジ及び円弧のエッジについて、これが２値化されたときの両者の関係を考察する．また一般の撮像装置で画像入力する際のノイズの影響についても言及する．

２．１　ディジタル測長精度の向上のための縦長画素の採用．

２．１．１　２本の走査線を監視する測長法

第３、４章に述べる手順に共通する精度向上の概念は縦長画素の採用である．手順の厳密な提示は当該の章で行うのでここでは簡単にその概要を示し、画素を縦長に採ることによる直線エッジの測長精度の向上の根拠を定性的に示す．

本論文が提唱するディジタル測長法は基本的に撮像装置からの映像信号に１～２走査線分のラインバッファを用いて（画像入力にはラスタースキャニング方式を前提としている）、４個の画素値を監視し，所定のアルゴリズムに基づき部分周囲長を積算していく形を採っている．この理由は次節でも触れるが処理速度の向上と処理に必要なメモリ量を少なく押さえ、且つ必要なハードウエアを可能な限り簡素化することを考慮したからである．

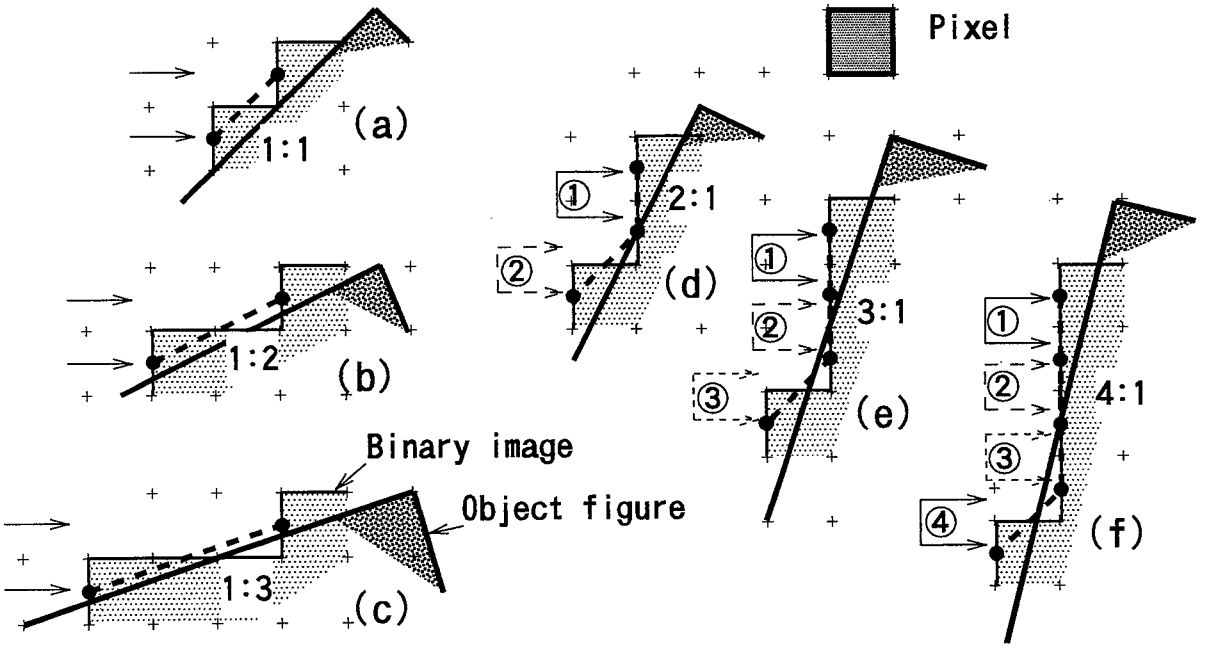


Fig.2.1 Relation between object figure, binary image and partial length

これらの条件は装置を実用化した際のコストの低減にもつながる．上記の理由で処理の対象となるデータはラインバッファ上の画素値と現在入力中の走査線上の画素値である．すなわち２つの走査線上の画素値を元に部分輪郭長が積算される．

図２．１に示すのは傾きの異なる６種（(a)～(f)）の直線エッジが２値化されたパターンについて，第３，４章で述べる手順による測長の様子を示している．傾きはm:nの比の形で表示しているがこの場合，縦方向：横方向の順で記している．今後このように表す．

図中(a),(b),(c)は元となる平面図形のエッジの傾きが45°以下の場合、(d),(e),(f)は45°以上の場合である．薄くハッチングした部分は元となる直線エッジ（太い実線）が２値化されて出来たパターンである．太い破線で示されているのは積算される部分長である．(a),(b),(c) はそれぞれ図形エッジの傾きが1:1、1:2、1:3の場合でこのとき積算される部分長は元の図形エッジの長さと全く同じである．また(a),(b),(c)のように比の縦方向に１が含まれ、横方向にnである場合、その部分長は$\sqrt{1+n^2}$と評価される．これは傾きが４５°以下の場合で且つ傾きが整数比でない、いわゆる有理数になるときにも２．３節での考察により、２値化した場合に生じるパターンは1:nと1:n+1の２種類になることが判明しているので理論的に測長値が求まる．その誤差は第３章の議論により比較的少ないことが判明する．

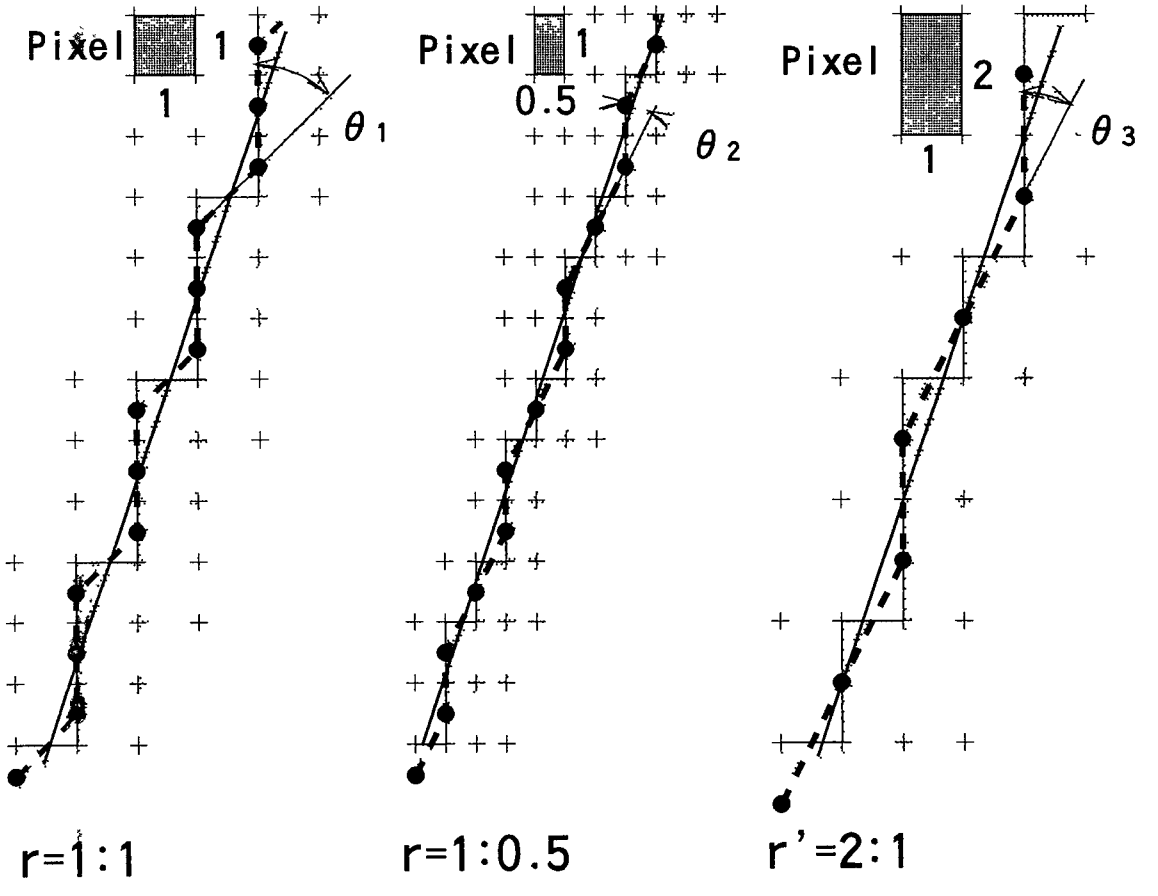


Fig.2.2 Accumulated partial length on square and rectangular Pixel

これに対して元となる直線エッジの傾きが2:1、3:1、4:1 、つまりn:1となる(d),(e),(f)

では２ラインずつの走査がそれぞれ2,3,4 回行われて（①，②，③，④で示す）部分長が積算される．図からわかるように積算される部分長は元の直線エッジの長さより大きくなり、その値はn-1+$\sqrt{2}$となる．結果として＋の誤差の発生が予想される．矢印の直線の対の種類は同時に走査される走査線の対を表しており、実線が第一回目、破線が第２回目、点線が第３回目、一点鎖線が第４回目を示している．

２．１．２　縦長画素の採用による測長精度の向上

図２．２は画素を縦長に採る場合の効果について示している．上記の如く傾きが45°より大きい直線について部分長の＋の誤差が大きいことを述べたのでこのケースについて議論する．図では傾きが3:1（画素上で）の直線エッジが２値化され、測定される部分長（太い破線）を示している．左図は画素の縦横寸法比rが1:1のときの様子を示している．中図は撮像装置の水平分解能を倍にし画素の縦横寸法比rが1:0.5＝2:1のときの様子を示している．右図は撮像装置の縦分解能を落とし画素の縦横寸法比rが2:1のときの様子を示している．

ここでジグザグに積算されていく部分長の傾きは左図でθ_1、中図でθ_2、右図でθ_3である．画素を表す升目の対称性により$\theta_1 > \theta_2 = \theta_3$ である．明らかに中図、右図の方が部分長積算値としては近似度が高い．第３章では主に右図の方式による測長を取り上げている．

２．１．３　走査線を飛び越して等価縦長画素を実現する方法

前項で述べた方法は撮像装置或いはサンプリング機構そのものの変更を必要とする．ここで述べる方法は正方形画素の映像信号を発生する撮像入力装置をそのまま利用して等価縦長画素を実現し測長精度を上げる試みである．

等価縦長画素を実現するため注目する走査線をいくつか飛び越して（スキップして）２本ずつ見ながら処理を進めていく．これは比較的ソフトウエア処理に向いている．

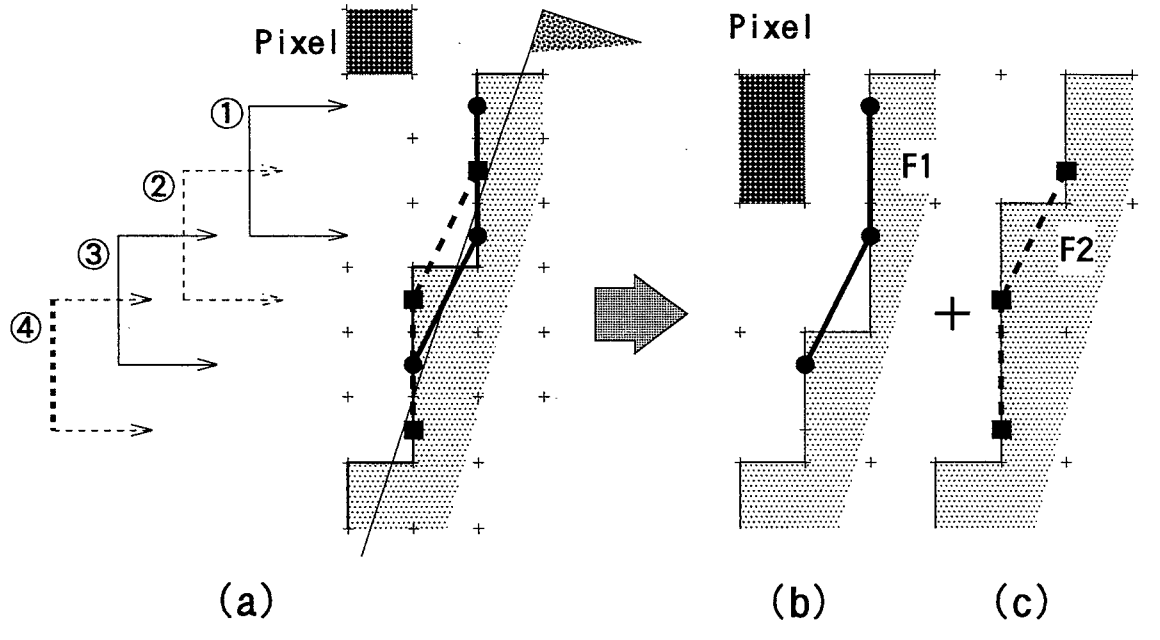


Fig.2.3　1 line skip scanning

その理由は通常の画像入力ＡＤ変換器は計算機のメモリ内に正方形画素としてデータを取

り込むからである．第４章ではこのケースについて考察を加えている．本項でも計算機メモリ内に２次元的に配置された結果としてのデータを対象として議論を進める．しかし走査線用のラインバッファを２個以上用意すればハードウエア上で実行することももちろん可能である．

図２．３(a)は走査線を一本飛び越して、２つの走査線上の画素値をもとに部分長を積算していく様子を示している．このように１走査線分だけ飛び越して（スキップs=1）走査して部分長を積算する．画像フィールドの右端に達したら１走査線だけずらし同様の操作を行う．図では奇数番目の走査に対応して求まる部分長を実線で、偶数番目の走査で求まる部分長を破線で示している．画素はこの時点では正方形である．一つの走査線を２回ずつ見ていくことになり最終的に求まった値は真の測長値の２倍となるのでこの値を２で除する必要がある．

さてこのような飛び越しを用いる走査は図２．３(b),(c)に示すように画素の寸法が2:1の二つの画像フィールド上での測長と等価である．つまり奇数番目の走査で求まる部分長は画像フィールドF_1上での測長値であり、偶数番目のそれは画像フィールドF_2上での測長値である．ここで画素寸法が2:1と縦長であるから２．１．２項で述べた縦長画素採用の効果が現れ測長精度が向上することが予測できる．

２．２　画像データ入力との並行処理．

コンピュータ画像処理において原始入力となる画像は本来，位置的に連続した２次元平面上で，同様に連続した濃度分布により構成された一つの集合体である．この画像をもとに具体的に計算機を用いて様々な加工，変換処理を加え観察者にとって有益な情報を取り出す操作をコンピュータ画像処理と定義してよいであろう．この過程において画像データそのものはいくつかの形態をとる．

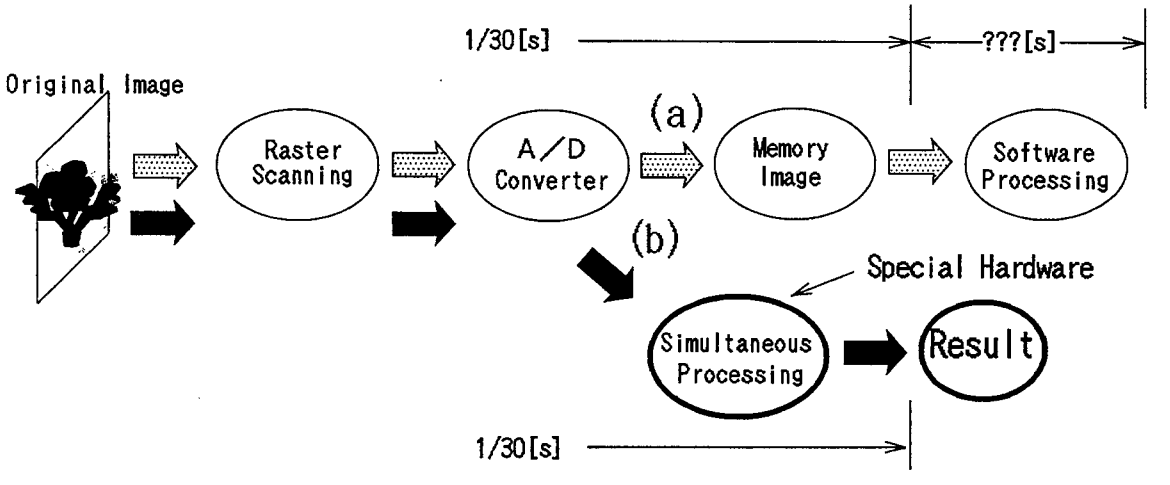


Fig.2.4 (a)Conventional processing and (b)Simultaneous processing

撮像にラスタースキャニング方式を用いることを前提とすれば，まず最初の撮像の段階では画像は時間，空間的にサンプリングされて時間的に直列のアナログ信号に変換される．

このアナログ信号はその濃度に対応する一定のレベルで切り分けられてディジタル化される．この時点で画像はコンピュータ画像処理の対象となるデータとなる．

”一般的には”このデータは計算機内のメモリ上に，入力される順序に従って規則的に配置され，一画面分の入力処理が終わってから初めてソフトウエア処理の対象としての形態を整える．”一般的には”とつけ加えた理由は本研究におけるディジタル測長手順が，画像データが計算機のメモリに１画面分読み込まれる以前から処理を開始し，時間直列に入力されてくる信号そのものから直接的に部分長を測長累算していくという”一般的でない”方法を採っていることがその特徴となっているからである．但しこの場合１～２走査線分のバッファメモリは最小限必要である．更に当然のことながら専用ハードウエアの使用を前提としている．通常の撮像装置の場合，一画面分のデータの取り込みには1/30[s]の時間を要するので，この間，計算機が待ち状態になる．これは実時間性を必要とするシステムでは大きな無駄時間である．第４章で述べる”スキップ２線走査法”では正方形画素を対象として専用ハードウエアを用いず，メモリ内に一旦取り込んでからのソフトウエア処理として述べているが画像データの入力作業と並行して処理を進める前述の考え方は基本的に共通しており、画像入力との並行処理を適用することはこの場合も可能である．逆に言えば専用ハードウエアを用意することが困難な場合にも適用可能なアルゴリズムと言える．

２．３　直線及び円弧の２値化の過程

ディジタル画像処理法は，溶接欠陥の非破壊検査，金属組織の解析，粉体の粒度観察など様々な分野で使用される有効な手法である．その際，図形の特徴パラメータ抽出の為の前処理として，入力されたイメージデータの２値化が行われる．２値化後のイメージに，それぞれのパラメータ抽出に固有のアルゴリズムが適用され目的の結果を得る．２値化イメージを微視的に見ると，その輪郭は画素寸法を単位とする階段状となっており，対象図形の移動により複雑に変化する．従ってアルゴリズムの性能を定量的に把握するには，図形を１画素寸法の範囲で微小移動，回転させたときのデータを数多く与えて評価する必要がある．このためのイメージデータとして，撮像装置から入力されたデータを用いたのでは電気的ノイズ，光学的な歪の影響が大きく，十分な評価ができない．このことはアルゴリズムの精度が比較的高い場合，特に問題となる．そこで，均一な感度特性，整列した感光素子の配列上に，幾何学的に完全な図形を重ねた理想化状態での，２値化イメージの作成が必要とされる．

本節では計算機のメモリ上に，理想化状態での幾何学的図形の２値化イメージを形成する問題ついて述べ，映像信号上に電気，光学的ノイズが重畳したときの問題，画素が長方形のときの２値化パターンについて議論している．対象図形としては直線輪郭を持つ矩形と，円弧を取り扱い，個々の画素値の”1”/”0”は図形が画素を覆う面積をもとに判断した．直線輪郭について，元となる直線エッジの傾きと，得られた２値化イメージの輪郭に生じる階段状のステップの種類及び個数との関係を示す式を導いた．円弧について図形輪郭と境界画素の関係を考察し，２値化メモリイメージを得る手順をフローチャート形式に示した．

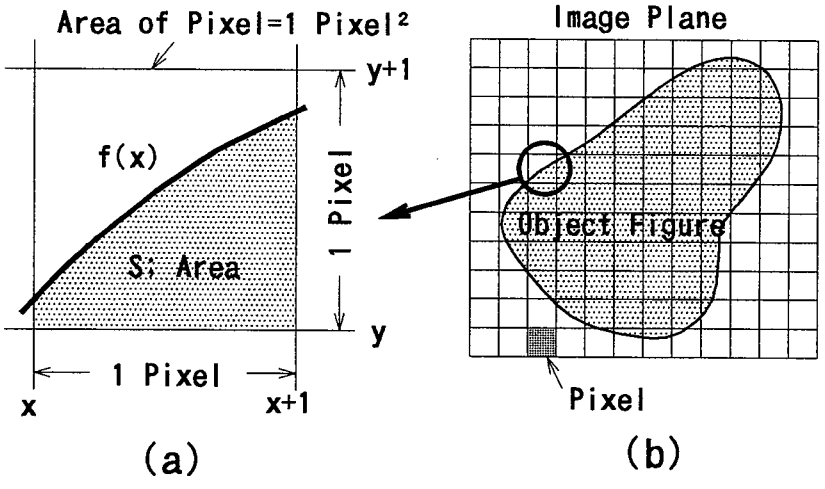


Fig.2.5　The pixel covered by figure

２．３．１　図形の２値化に関する解析

（１）画素値の"1"/"0"の判定基準

計測用２次元イメージセンサとしてＣＣＤイメージセンサが多用される．感光部はフォトダイオードの２次元アレイで構成されており，並列コンデンサに蓄積した電荷をＣＣＤトランスファチャンネルを用いて転送し読み出す．２次元アレイを正方形配列として理想化し，これに図形のイメージが重なった状況を図２．５に示す．感光素子は面積を持っているのでその発生起電力 V は図形が素子を覆っている面積 S に比例する．すなわち

$$V \propto S = \int_{x}^{x+1} \int_{y}^{f(x)} dydx$$

V は0からV_{max}の値をとる．これを２値化するにあたっては，いき値をどのように設定するか多くの議論[2]があるが，ここでは周囲の画素の値とは独立して決定することにし，50%値を採用する．すなわち

$S < 0.5$ [Pixel²] のとき　"0"
$S \geq 0.5$ [Pixel²] のとき　"1"　　---(2.1)

と判定することにする．図形イメージが当該画素を完全に覆っていればその画素の値は"1"であり，部分的に覆われている場合，すなわち図形のエッジが画素を横切るときが問題となる．本文ではこのような情況を取り扱う．

（２）直線エッジの２値化

対象図形が，両端の影響が無視できるような長い直線エッジの場合について，これが２値化されたときに生じるパターンについて考察する．この場合，式(2.1)の面積 S は容易に推定でき，従って画素値の決定も容易である．すなわち図形のエッジが画素中心の上か

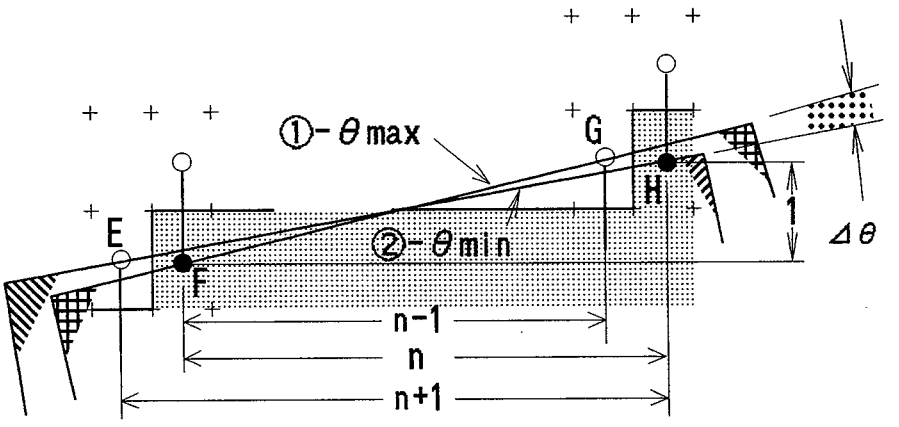


Fig.2.6 Binary image and original straight lines (1:n)

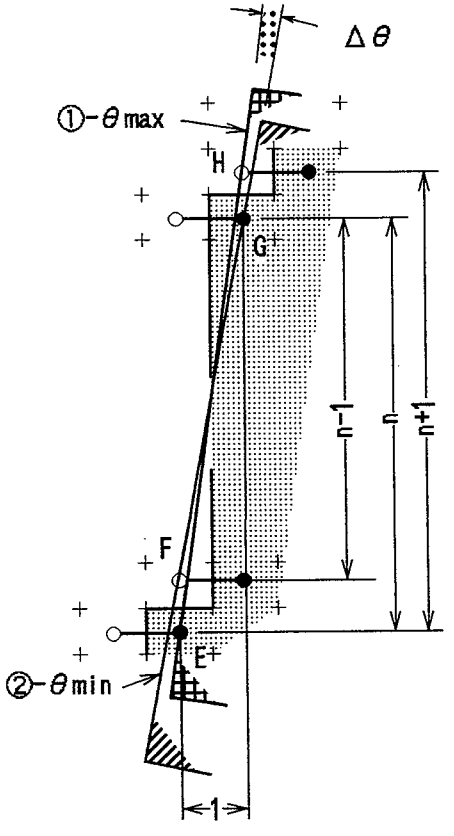


Fig.2.7 Binary image and original straight lines (n :1)

下か、いずれを通るかで決定される．真上のときはここでは" 1 " と判定することにする．傾きが整数比 1:n （ n ;整数，比の一方に 1 を含む）の直線エッジが 2 値化されると 1 :n の階段状のステップを生じることは明らかである．ここで 1:n は Y 方向: X 方向= 1 :n を意味する．今後このように表す．次に一般的な有理数の傾き $\theta = tan^{-1}(b/a)$，比で表せば b:a （ b, a; 正の整数)の直線エッジがどのように 2 値化されるかを検討する．この問題に関して次の補題(1),(2) が成立する．証明は付録参照のこと

補題 (1) 2 値化後のイメージが 1:n のステップ状になるとき($\theta \leq 45°$)，もとの直線

エッジの存在範囲に関して、直線エッジの傾きを $\theta = tan^{-1}(b/a)$ とすれば　（$b \leqq a$；正の整数、$n \geqq 1$，図２．６参照）

$$n-1 < a/b < n+1 \qquad \text{---(2.2)}$$

補題（2）　２値化後のイメージが n:1 のステップ状になるとき（$\theta > 45°$），もとの直線エッジの存在範囲に関して、直線エッジの傾きを $\theta = tan^{-1}(b/a)$ とすれば　（$b > a$；正の整数、$n \geqq 1$，図２．７参照）

$$n-1 < b/a < n+1 \qquad \text{---(2.3)}$$

補題(1)を使って次のことが証明できる．

定理(1)　傾き b:a （$b \leqq a$；正の整数，　$\theta \leqq 45°$）の直線エッジの一部が２値化されたときのイメージは 1:n と 1:n+1 の２種類のステップで構成される．

但し n は a を b で除したときの整数部である．このとき，それぞれのステップの個数 p, q は

$$p = (n+1)b - a \qquad \text{---(2.4)}$$

$$q = a - nb \qquad \text{---(2.5)}$$

証明は付録参照のこと

定理(1)と同様に a, b が $a < b$ の場合については補題(2)を用いて，次の定理が証明できる．

定理(2)　傾き b:a（$b > a$；正の整数、$\theta > 45°$）の直線エッジの一部が２値化されたときのイメージは n:1 と n+1:1 の２種類のステップで構成される．

但し n は b を a で除したときの整数部である．それぞれのステップの個数 p, q は

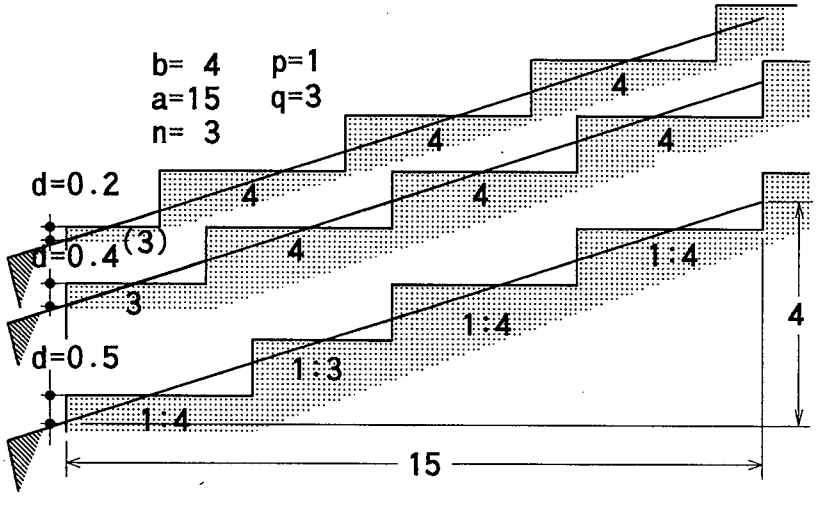


Fig.2.8　Example of binary image for straight line edge with slope of 4:15

$$p = (n+1)a - b \qquad \text{---(2.6)}$$

$$q = b - na \qquad \text{---(2.7)}$$

証明は付録参照のこと

以上の議論は傾きが整数比で表わされる場合、すなわち有理数について行った．現実の

傾きの比として無理数であっても近似的に有理数に置き換えることが可能であるので実用的にはこれで十分である．

図２．８ に b:a=4:15 の例を示す． $3 < 15/4 = 3.75 < 4$ であるから n= 3 であり式（2.4)、(2.5）より p=1, q=3 となる．図の上方には直線のエッジが１画素の範囲で微小量平行移動したときを示している．（ d=0.5, 0.4, 0.2 Pixel, d は画素境界からの偏位である．）

いずれも p=1, q=3 でありステップの現れる位置が異なるだけである．d=0.2 については 1:3 のステップが左下と右上に分離している．

（３）円弧の２値化

直線に比較して円弧を規定するパラメータは多い．半径 r，中心座標 x_c, y_c 始角 θ_1, 終角 θ_2の５つがある．円弧の２値化については直線の場合の様に，ある程度の長さをと

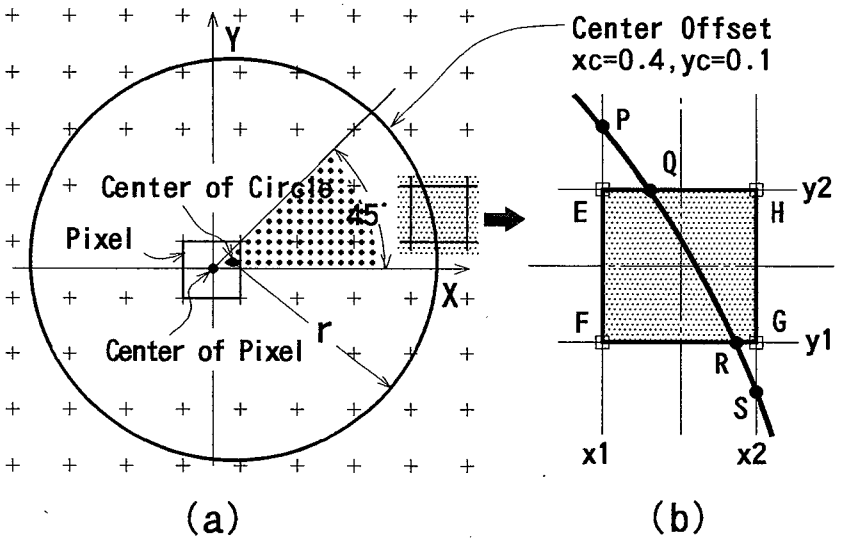


Fig.2.9 A circle and arc. (a) Circle on image field, (b) Arc on one pixel

れば同じパターンの繰り返し的な処理，が不可能である．すなわち定理(1),(2)のような一般化された規則は見いだせない．そこでr, x_c, y_cが決定した円に対して，画像フィールド上の個々の画素について式(2.1)に従って"1"/"0"を決定することになる．

画素全体が円の内部に含まれていれば"1"である．従って円弧が画素を横切るときを検討すればよい．折り返しと回転を利用すると解析すべき範囲は第１象現の $0° \leq \theta \leq 45°$ の範囲で良いことがわかる．図２．９（ａ）はこの様子を示している．円は

$$(x-x_c)^2+(y-y_c)^2=r^2 \qquad \text{---(2.8)}$$

で表され，中心オフセット量 x_c, y_c は，１画素ずれれば原点がずれたものと等価であるから

$-0.5 \leq x_c \leq 0.5$

$-0.5 \leq y_c \leq 0.5$

を考えればよい．但し原点は画素中心である．図２．９（ｂ）は円弧と画素の交差部分を示している．E, F, G, H は画素の頂点である．P, Q, R, S はそれぞれ x_1, y_2, y_1, x_2 軸との交点でその座標は，次のとうりである．

$P(x_1, y_p)$; $(x_1, \sqrt{r^2-(x_1-x_c)^2}+y_c)$, $Q(x_Q, y_2)$; $(\sqrt{r^2-(y_2-y_c)^2}+x_c, y_2)$

$R(x_R, y_1)$; $(\sqrt{r^2-(y_1-y_c)^2}+x_c, y_1)$, $S(x_2, y_S)$; $(x_2, \sqrt{r^2-(x_2-x_c)^2}+y_c)$

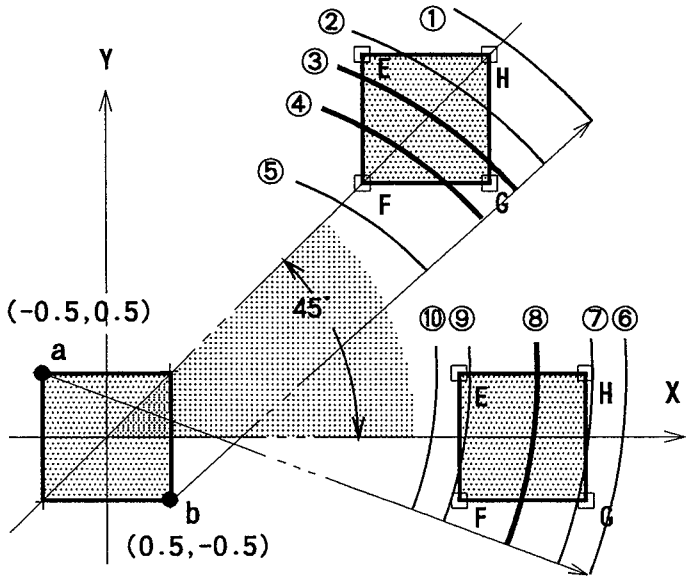


Fig.2.10 Behavior of arc line on pixels

円弧と画素が交差するパターンを，頂点 E,F,G,H との関連において分類すると，図２．１０①②--⑩のようになる．①--⑤は円の中心が，原点画素の右下隅 b 点(0.5,-0.5)に

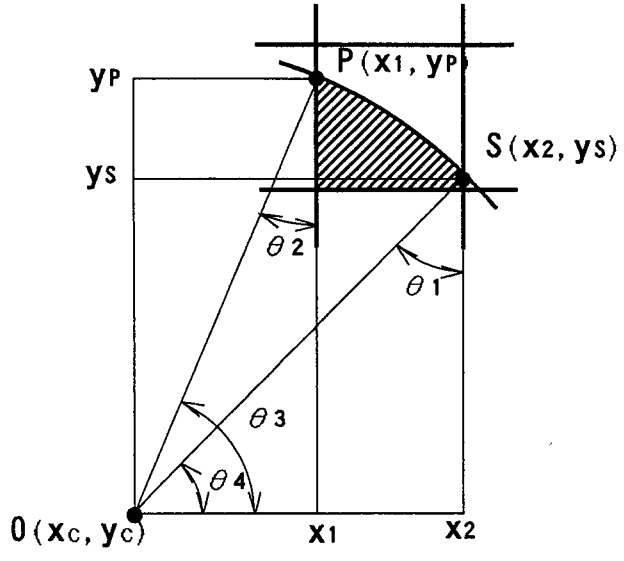


Fig.2.11 Intersection of arc and pixel (case 3)

あるときの，⑥--⑩は同じく左上隅 a 点(-0.5,0.5)にあるときの様子を示している．x_c, y_c, rのすべての組み合わせの任意の画素に対するパターンはこの中に尽くされている．①,②,⑥,⑦については画素の値は "1"， ⑤,⑨,⑩は "0" であることは明らかである．⑧では円弧が画素の右辺を横切る場合があるが，このとき画素の値は "1" である．③,④,及び前述のケースを除いた⑧については積分により面積を求めて判断する．

③の場合，当該画素が円の内部に含まれる面積 S_3は図２．１１のように記号を定めると

$$S_3=\int_{X_1}^{X_2}\int_{y=y_1}^{y=\sqrt{r^2-(X-X_c)^2}+y_c} dydx \qquad \text{この積分は付録を参照して}$$

$$S_3=\frac{1}{2}r^2\{\cos^{-1}\frac{X_1-X_c}{r}-\cos^{-1}\frac{X_2-X_c}{r}$$

$$+\frac{1}{2}\{(X_2-X_c)(y_s-y_c)-(X_1-X_c)(y_p-y_c)\}-(y_1-y_c)(X_2-X_1) \qquad \text{---(2.9)}$$

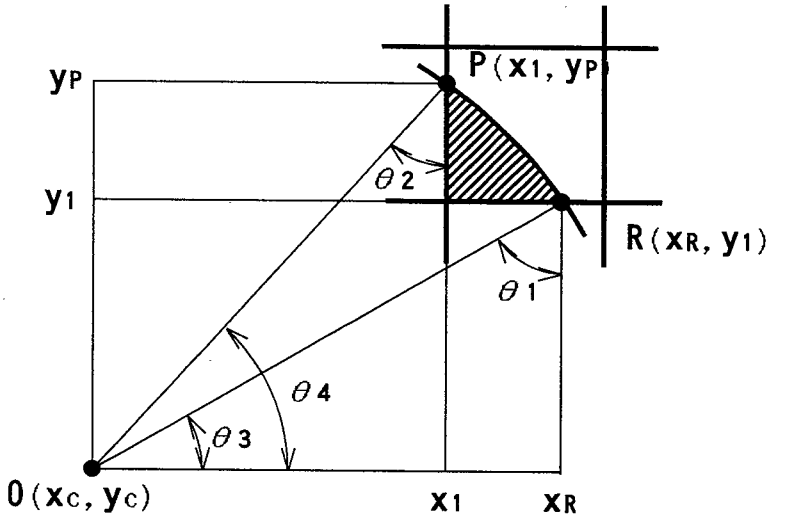


Fig.2.12　Intersection of arc and　pixel (case 4)

④の場合，当該画素が円の内部に含まれる面積 S_4は図２．１２のように記号を定めると

$$S_4=\int_{X_1}^{X_R}\int_{y=y_1}^{y=\sqrt{r^2-(X-X_c)^2}+y_c} dydx \qquad \text{この積分は付録を参照して}$$

$$S_4=\frac{1}{2}r^2\{\cos^{-1}\frac{X_1-X_c}{r}-\sin^{-1}\frac{y_1-y_c}{r}\}$$

$$+\frac{1}{2}(X_Ry_1-X_Ry_c-X_cy_1+X_cy_c-X_1y_p+X_1y_c+X_cy_p-X_cy_c+2y_cX_R-2X_1y_c-2y_1X_R+2y_1X_1)$$

$$=\frac{1}{2}r^2\{\cos^{-1}\frac{X_1-X_c}{r}-\sin^{-1}\frac{y_1-y_c}{r}\}$$

$$-\frac{1}{2}\{(X_1-X_c)(y_p-y_1)+(X_R-X_1)(y_1-y_c)\} \qquad \text{---(2.10)}$$

⑧の場合，当該画素が円の内部に含まれる面積 S_8は図２．１３のように記号を定めると

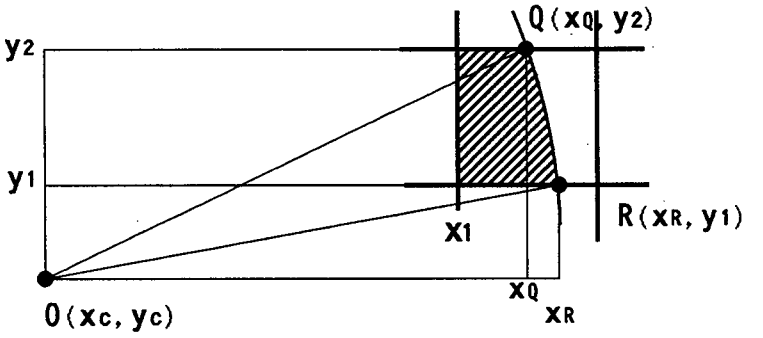


Fig.2.13 Intersection of arc and pixel (case 8)

$$S_8=\int_{y_1}^{y_2}\int_{X=X_1}^{X=\sqrt{r^2-(y-y_c)^2}+x_c} dxdy$$ この積分は付録を参照して

$$S_8=\frac{1}{2}r^2\{\sin^{-1}\frac{y_2-y_c}{r}-\sin^{-1}\frac{y_1-y_c}{r}\}$$

$$+\frac{1}{2}\{(y_2-y_c)(x_Q-x_c)-(y_1-y_c)(x_R-x_c)\}+(x_c-x_1)(y_2-y_1) \quad ---(2.11)$$

となる.

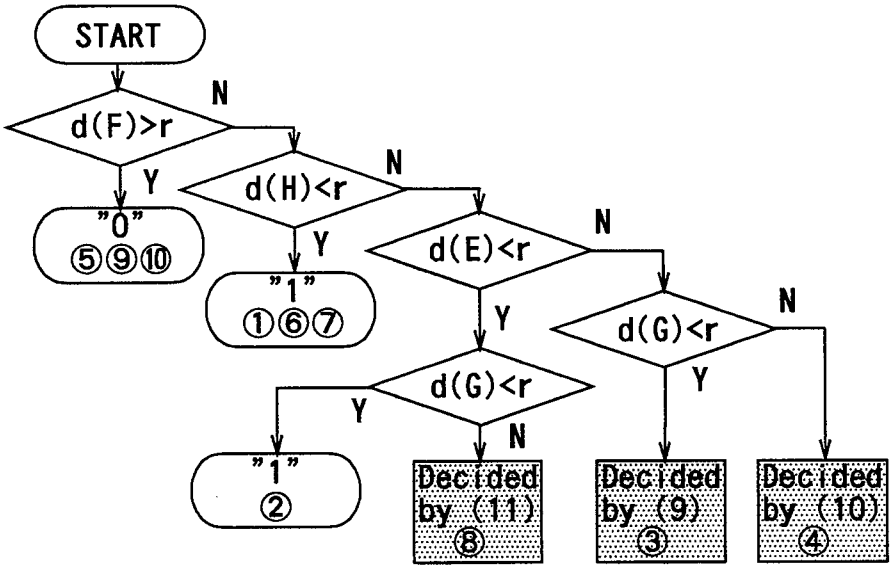


Fig.2.14 Procedure for determination of logical value "1" or "0" of pixel

そこで画像フィールド内任意の点 P(x, y)と円の中心座標との距離を計算する関数 d(P)を次のように定義すれば

$$d(P)=d(P(x,y))=\sqrt{(x-x_c)^2+(y-y_c)^2} \quad ---(2.12)$$

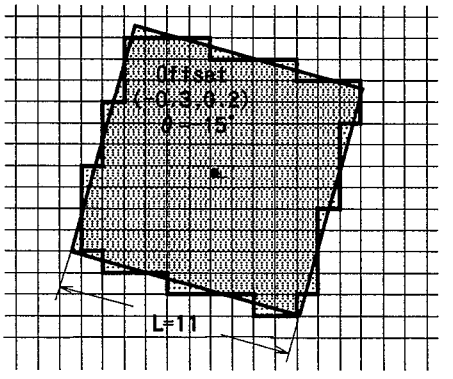


Fig.2.15 Image formation of square

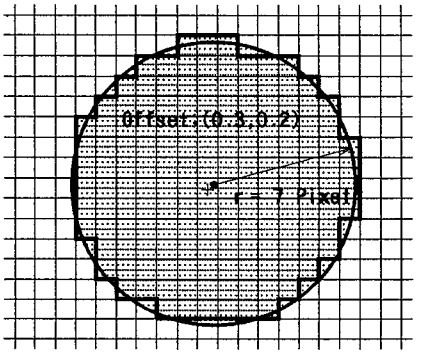


Fig.2.16 Image formation of circle (Large circle)

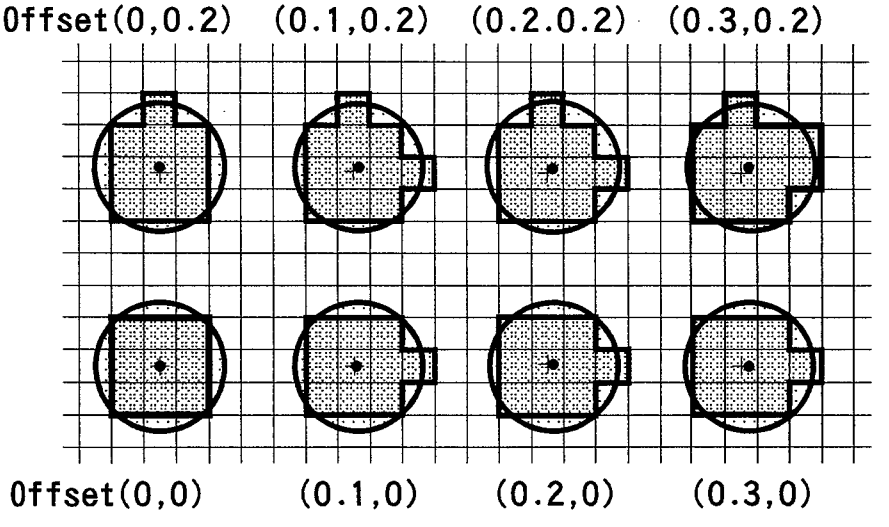


Fig.2.17 Image formation of circle (Small circle)

式(2.9)～(2.12)から，図２．１４のフローチャートにより全ての画素の"1/0"が決定できる．なお図中の(9)～(11)は式(2.9)～(2.11)に対応する．⑧で画素の右辺を横切る場合，式(2.11)は画素が円弧の内部に含まれる面積を表すものではないが，式の値は0.5を越えるのでフローチャート上ではこの中に含めることができる．

るのでフローチャート上ではこの中に含めることができる.

2．3．2　計算結果

（1）導出した公式による2値化パターン

図2．15は正方形を2値化した例である．但し頂点については特別に求積処理をしている．正方形の中心は画素中心から(-0.3,0.2)だけオフセットしている．また正方形の寸法は一辺が11 Pixel であり，15° だけ右に傾斜させている． 図2．16は図2．14の手順に従って円弧を2値化した例で，比較的半径が大きいときのものである．半径は7 Pixel，オフセットを(0.3,0.2)だけ与えている．図2．17は微小円弧について，オフセットによる変化を示している．左下隅がオフセット0で X 方向へオフセット0.1 Pixelきざみで，Y 方向へ0.2きざみでシフトしながら表示している．

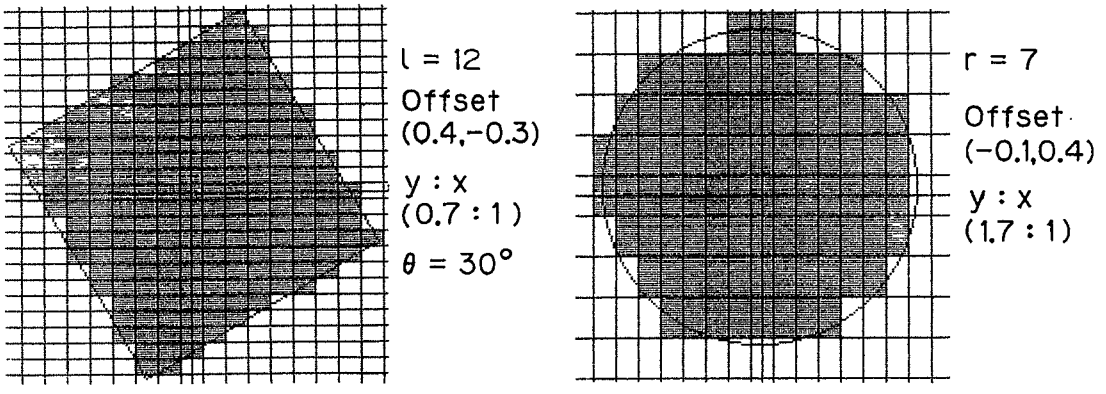


Fig.2.18　Two kind of image formation on rectangular pixel

（2）長方形画素による2値化パターン

第3章でディジタル測長に関する隣接2線走査法について述べる．この手法は，画素を長方形，特に縦長に採ったとき測定精度が向上することが明らかになっている．この意味で長方形画素上の2値化パターンが必要となる．2．3．1項の（2）及び2．3．1項の（3）で述べた2値化の手順は簡単な変更で長方形画素の場合にも適用可能である．図2．18はこれを示している．（a）は画素寸法を Y方向 :X方向を 0.7:1 に採って一辺の長さ12 Pixelの正方形を2値化したパターンである．但し正方形の中心と画素中心のオフセットを(0.4,-0.3) Pixel 与え，傾きは30°である．（b）は画素寸法を Y方向:X方向を 1.7:1 に採って半径 r =7 の円を2値化している．但し円の中心と画素中心のオフセットは(-0.1,0.4) Pixel である．

2．3．3　2値化形状計測に於ける誤差発生要因の分析

（ここでは *S*-面積、*G*-重心、*L*-周囲長、*F*-形状係数を対象とする．）

誤差発生要因は次の3つに分類される．

(a)　形状信号（アナログ）そのものに含まれる誤差

(a-1)　光学系に関するもの－－－レンズ、撮像素子の物理的歪み

(a-2)　電気的ノイズ、照明の不均一、ちらつき

(c)　2値化された形状データより S, G、L、F を求めるアルゴリズムに起因する誤差

これらは誤差発生の性質が根本的に異なる．(a-2)は統計的にバラツキを持ったノイズに起因する．従って複数回の測定によって得られるデータはその精度に応じた一定の範囲に分布し、平均をとることによって，回数の平方根に比例して精度が上がる．(c)に関しては形状データが一定なら結果は手順により一意的である．つまり誤差があったとしても測定のバラツキは0である．

最後に(b)について述べる．図形の形状が一定でも画素との位置関係により2値化後の図形の形状が異なるという問題である．図形が小さく画素の大きさとほぼ同程度か、あるいはそのような細かい模様の場合に問題が生じる．さらに一般的に言えば図形の空間周波数がサンプリング周波数に近いときに起こる．

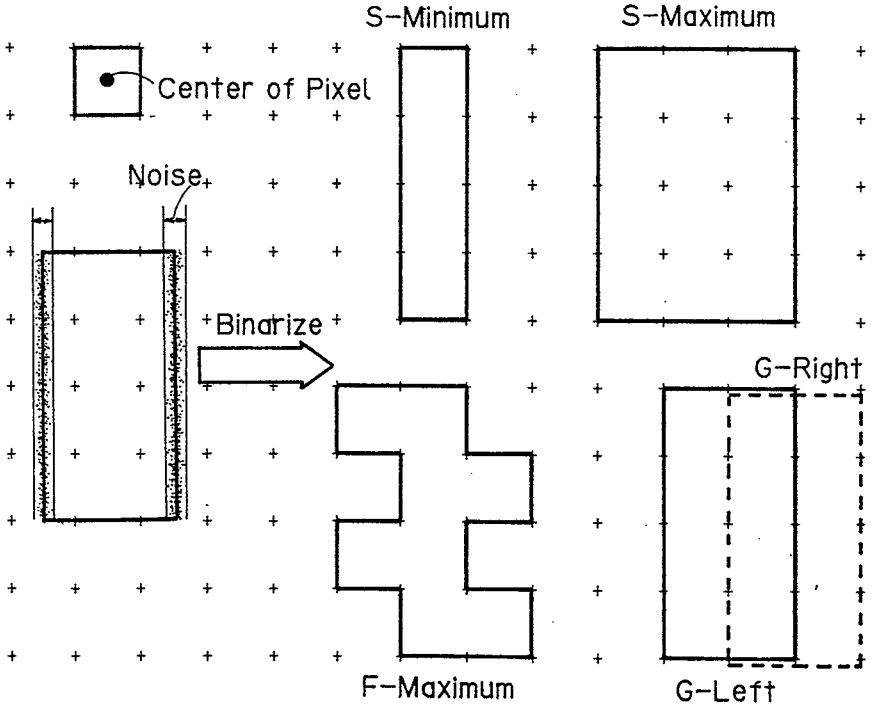


Fig.2.19　Extreme case of converting to binary image

これらの誤差発生の極端な例を 図２．１９に示す．左の長方形が画素の境界にかかり，2値化されて右の4種の2値化パターンを生じている．

これらを見ると直線で囲まれた幾何学的な図形の場合に問題が多い．図形輪郭が垂直な（水平な）場合，それが画素中心の左右（上下）いずれの側にあるかで面積が跳躍的に変化する．つまり長さ n 画素の直線エッジは潜在的に n 画素分の誤差発生の可能性を持っている．自由曲線で囲まれた一般的な図では平均化されて極端な誤差は起こり難いと考えられる．これは実用的には有利である．形状計測を高速に行う場合、数回測定して平均をとる余裕はない．対象が幾何学的図形のとき，最大と最小の差を小さくするという意味では平均値0のノイズを重畳することも考えられる．つまり分散は一定でも分布の形がガウス曲線に近くなる．

２．４　結言

本章で明らかにしたことを以下に箇条書きにして要約する

(1)使用メモリが少なく且つ高速化が可能な２本の走査線上の画素を観測しながら輪郭測長を行うアルゴリズムの概要を示した.

(2)画像フィールドを構成する画素画素として縦長の画素を採用することにより上記アルゴリズムによる測長精度が向上することを定性的に示した.

(3)専用ハードウエアを用いて、画像処理データの入力と平行して処理を行うことにより待ち時間無しに結果が得られることを示した.

(4)任意の傾きの直線エッジが２値化されたときに生じる単位ステップの種類は１または２種類であることを示し，その個数を表す式を誘導し，式が厳密に正しいことを数学的に証明した.

(5)画素を覆う図形の面積を基準にとり，円弧エッジの２値化パターンを求める手順を明らかにした.

(6)原図形に光学的，電気的ノイズが重畳した際の２値化パターンの構造について検討し，図形輪郭が直線のときに比し，曲線で囲まれているときの方が測定値にばらつきを生じにくいことを明らかにした.

第３章　縦長画素を用いたディジタル測長

３．１緒言

第１章においてディジタル測長という言葉を定義した．すなわち画像データの濃度等高線に沿った，２点間の長さである．この長さはそれが閉じているときに周囲長という言葉で表現される．図形の内部に空洞が存在するときこれを周囲長の一部として扱うか否かは異論があると思われるが，溶接部の欠陥の検査などでは，欠陥の度合いを表す係数として形状係数を用いるならば”周囲”に含めても良いと思われる．本章ではこのような意味で周囲長を用いる．その他の図形形状パラメータとして面積，重心，形状係数などがある．これらのパラメータも金属組織解析の分野では非常に有効なパラメータである．本章では新しく開発した，ディジタル測長のアルゴリズム”隣接２線走査法”について，その原理の詳細を述べる．第２章では縦長画素の採用により測長精度が向上することを定性的に示した．ここではこの方法でもディジタル測長精度の向上が可能なことをシミュレーションと実験により証明する．また隣接２線走査法を実際のハードウエア上に組み込んで評価した結果について述べる．

ディジタル測長を行う場合，どこを始点，終点と考えるかについて曖昧さが発生する．このため，本文中の記述では，この２つが連続している場合，すなわち周囲長の問題として取り扱っている．しかしここでの議論は，本質的には図形エッジの２点間の測長に適用可能である．

３．２　隣接２線走査法の原理と縦長画素の採用によるディジタル測長精度の向上

３．２．１　あらまし

２次元形状計測に於て面積，重心，周囲長，形状係数は図形の特徴を表現する基本的パラメータである．例えば溶接継手部に発生する欠陥の種類をその形状係数から判別することができる．また金属組織解析の分野ではダクタイル鋳鉄の球状黒鉛の球状化率を，面積と形状係数で表現する．この中で面積，重心の測定は容易であるが周囲長測定は原図形の形状推定と関連して比較的複雑である．

従来の周囲長測定の方法としては　①メモリイメージ上で画像輪郭を追跡し画素の接続方向との関係において部分長さを割当て，これを積算する方式（$\sqrt{2}$　法，$\sqrt{2}$-$\sqrt{5}$　法，３画素ベクトル法，ランレングスから求める方法など）　②Hough変換などの写像変換を施した後，点の連続性を推定しこれから周囲長を求める方式などがある．前者はアルゴリズム次第では高い精度が得られる．後者はノイズなどによる輪郭の欠落に強い．しかしこれらのいずれの方法も処理に多大の時間を要する．ディジタルシグナルプロセッサや専用ハードウェアにより高速化しても本質的に方式としての限界が存在する．

筆者はこの処理時間の問題を解決し，且つ精度的にも十分な周囲長測定のアルゴリズム”隣接２線走査法”を考案した．２値化画像信号を時間直列に入力し，１走査線分のメモ

リとの間で部分輪郭長を生成し，連続的に積算する方式である．従来の方式に於けるメモリ参照のためのアドレスの不規則な後戻りがなく高速処理に向いておりハードウエアで処理する場合も非常にシンプルな構成となる．パイプライン化も容易で通常のビデオレート相当のスピードも不可能ではない．

さらに，本アルゴリズムによる周囲長評価の過程を解析した結果，２．１．２項で述べたように画素寸法を縦長にとることにより測定精度を大きく向上できることが判明した．ラスタスキャン方式で縦長画素を実現するには横分解能を上げるか縦分解能を落とす必要がある．この手順によれば総合分解能が落ちると考えられる後者の方法でも周囲長測定精度は向上することがわかった．

以上の測定法を２値化された図形についてシュミュレーションを行った結果，アルゴリズムの正当性及び縦横比の増大による精度の向上を確認することが出来た．

３．２．２　周囲長計測の手順

以下に”隣接２線走査法”の手順を説明する．

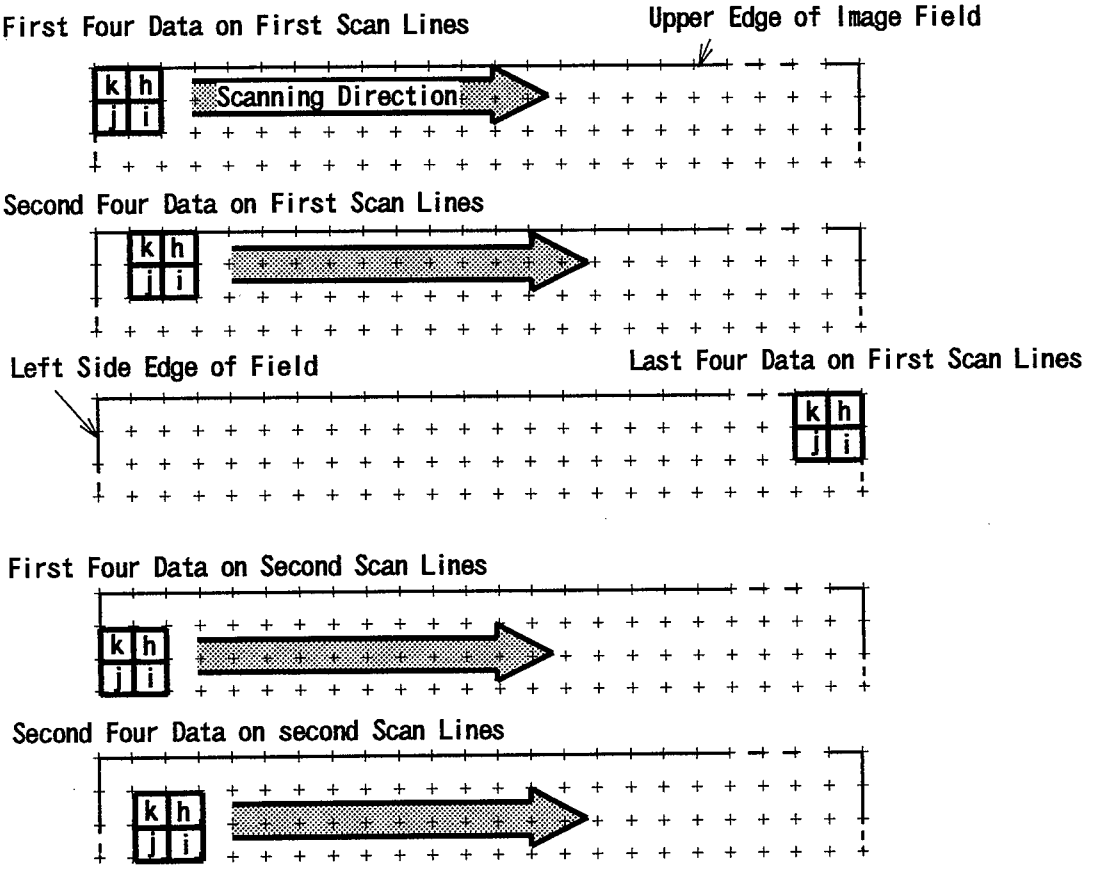


Fig.3.1　Scanning of image field (Raster scanning)

画像データのアクセス方法は基本的にテレビジョンのラスタスキャンと同じである．但

し，図３．１に示すように h, i, j, k で表される４個の画素を，同時に観測しながら進めていく．左上隅の４個から始めて右水平方向に１画素ずつずらして走査して行き，画像フィールドの右端に達したら，縦位置を１ラインだけずらして次のスキャンを開始する．h, i, j, kの４個の画素値は次に示す演算手順の入力データとなる．走査は画像フィールドの右下端に達すると終了である．この間に下記の手順により部分輪郭長を積算し１フィールドの終了と同時に計測を終らせる．

図３．２は手順の説明のための画素の対応関係を示している．i を今注目している画素，h は上の画素，j は注目している画素の一つ前の画素，k は上の画素の一つ前の画素である． 以下に手順の詳細を示す．

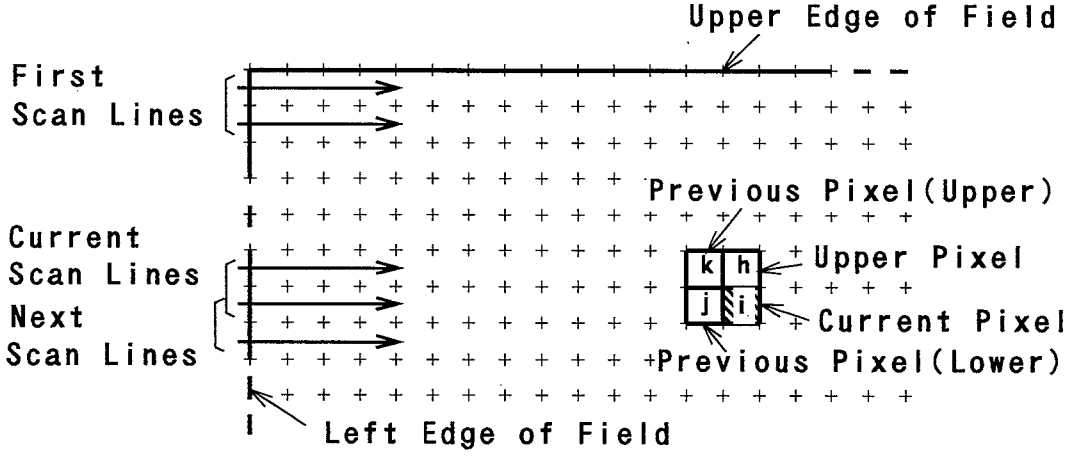


Fig.3.2 Scanning and pixels

「n = 0 とする．２値化後のメモリイメージに対して相隣る２つのライン上を左から右へスキャンする．上下対応する画素の値の排他的論理和が連続して"１"である画素の数を数えこの値を n とする．上下画素の排他的論理和が" ０ " で且つ，上下の画素のそれぞれ一つ前の画素との排他的論理和が，上下でいずれかが"１"である点（これを計算点と呼ぶ）に到達したら $\sqrt{1+n^2}$ を計算し累算部にたし込む．この後 n を０にクリヤする．前記の４個の画素値がすべて "0" か，すべて "1" なら n = 0 とする．画像フィールドの右端に到着したら１ラインだけ下へずらして同じ操作を行う．計算点は論理式で表すと $(\overline{h \oplus i}) \cdot (k \oplus h + j \oplus i)$="1" を満足する点である．」

以上の手順を" 隣接２線走査法" （Dual Lines Scanning method ）略してＤＬＳ法と呼ぶことにする．以後このように記す．

計算点のパターンは６個ある．これを 図３．３ に示す．図の上段と下段は互いに補の関係にあり，(a),(d)は図形エッジが垂直な左辺，右辺の長さ積算に対応する．(b)は上辺，(c)は下辺に対応する．同じく(e)は底辺の終端，(f)は上辺の終端に対応する．

図３．４は図形の上辺に於ける手順の実行例である．左から走査してきて①の時点まで４個の画素値はすべて "0" であったとする．①の時点で入力データとなる画素は h, i, j, k となる．ここでは h と i の排他的論理和が "1" であるから n が＋１されて n の値

は1となる．次に右に移動して②の点に来ると入力データとなる画素は h', i', j', k' となる．ここでは h' と i' 及び j' と k' の排他的論理和はいずれも "1" であり，前記同様に n は＋1されて n=2 となる．同様の演算が⑥の時点まで継続して n は増加する．⑦の所に来ると，このパターンは図3．3に示した計算点(b)のパターンに一致する．従ってここで $\sqrt{1+n^2}$，すなわち $\sqrt{1+6^2}$ が計算されて累算部にたし込まれる．同時に n は0にクリヤされる．

排他的論理和の変数の，順序交換が可能なこと，及び図3．3の上下のパターンが互いに補であるという性質から，図3．4で画素の "1","0" の値が反転した場合も全く同じ演算が行われることは明らかである．すなわち図3．4は図形の上辺部についての例であったが，画素値が反転した図形の下辺部についても同様に計算される．次に図3．4の図形の上辺部に凸部が存在する場合の例を図3．5に挙げる．

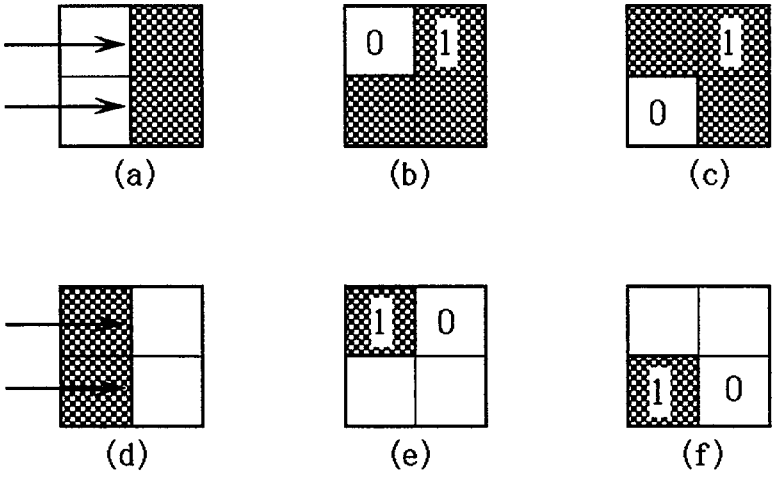


Fig.3.3　6 Patterns of calculating point

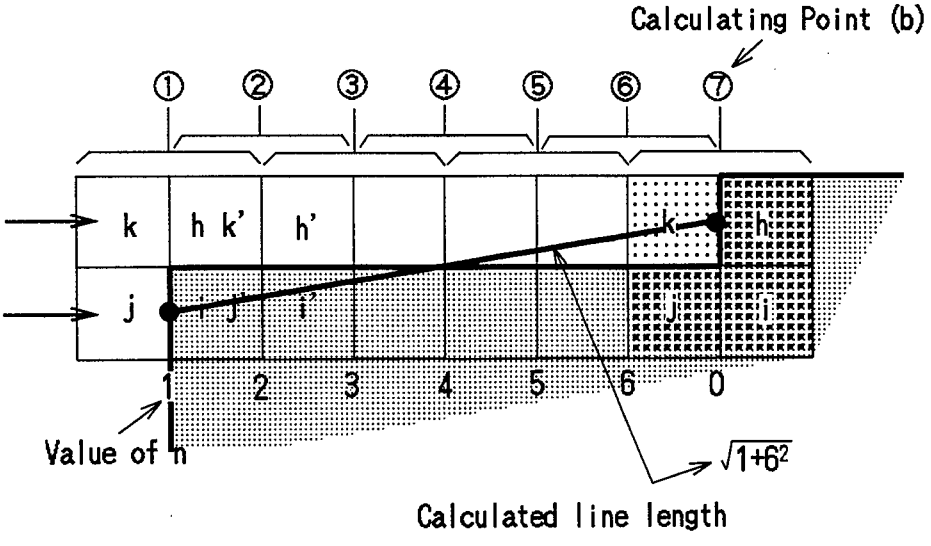


Fig.3.4　Example of calculation using DLS algorithm

図3．5では①から n の計数が始まる．②，③，④，⑤まで上下画素の排他的論理和が

"1" なので n の値が順次＋１されて行く．⑥は図３．３の計算点パターン(f)である．したがって$\sqrt{1+n^2}$ の $n=5$ で$\sqrt{1+5^2}$が計算されて累算器にたし込まれる．このとき累算される長さ$\sqrt{1+5^2}$を③と④の中間で折り返して（Turn down),図の実線の如く対応づけする．このようにすれば次の累算長との接続関係がスムースになる．この関係は図形の下部に凸部を有する場合にも同様に適用できる．

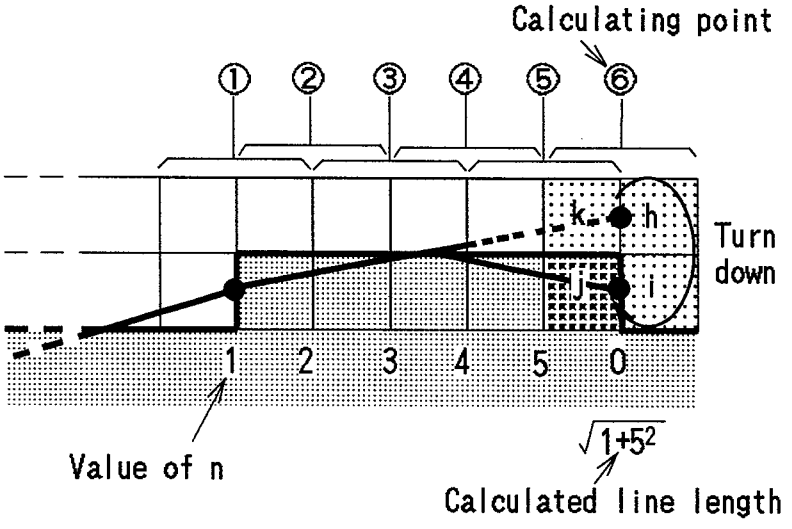


Fig3.5 Length calculation including 凸 part

図３．６は内部に空洞を持った図形に対する本アルゴリズムの総合的な実行例である．黒丸は計算点を示し黒丸間を結ぶ破線は $\sqrt{1+n^2}$ の長さを示している．この長さが累算部に積算される．図形が上下方向に凸の部分では，前述のごとく$\sqrt{1+n^2}$ を示す直線を中央で折り返して ”周囲” に対応付けする．空洞がある場合，空洞の上部は通常の図形の下部とみなして計算する．下部はその逆である．視野内に図形が複数個あれば 図３．６のように長さの対応付けがなされた図形が複数個でき，最終的には累算部に総周囲長が求まる．

上記の手順で，計算点における判断に必要なものは規則的に配列された，h, i, j, k ４アドレス分のデータである．そして n のカウントアップと判断を画素の数だけ繰り返す単純な構造であって，総データアクセス回数も従来の手法に比して非常に少ない．これらは高速処理とハードウエア化の容易さの条件に適合する．ハードウエアでリアルタイム処理する場合 i は現データ，j は１画素の遅延データとして h, k は１走査線遅延メモリにより得られるので１画像フィールド分のメモリを必要としない．また$\sqrt{1+n^2}$ の計算にはＲＯＭ(Read Only Memory)によるテーブルルックアップが利用できる．

３．２．３　単位ステップの長さ評価

（傾きが $1:n$ または $n:1$の時）

もとの図形が傾き $1:n$ の，端部の影響が無視できるような長い直線エッジを持つ場合，これが２値化されると $1:n$ の階段状のステップを生じることは明らかである．この部分の長さはＤＬＳ法によれば次のように評価される．ここで$1:n$ は　Y方向：X方向＝$1:n$ を意味する．

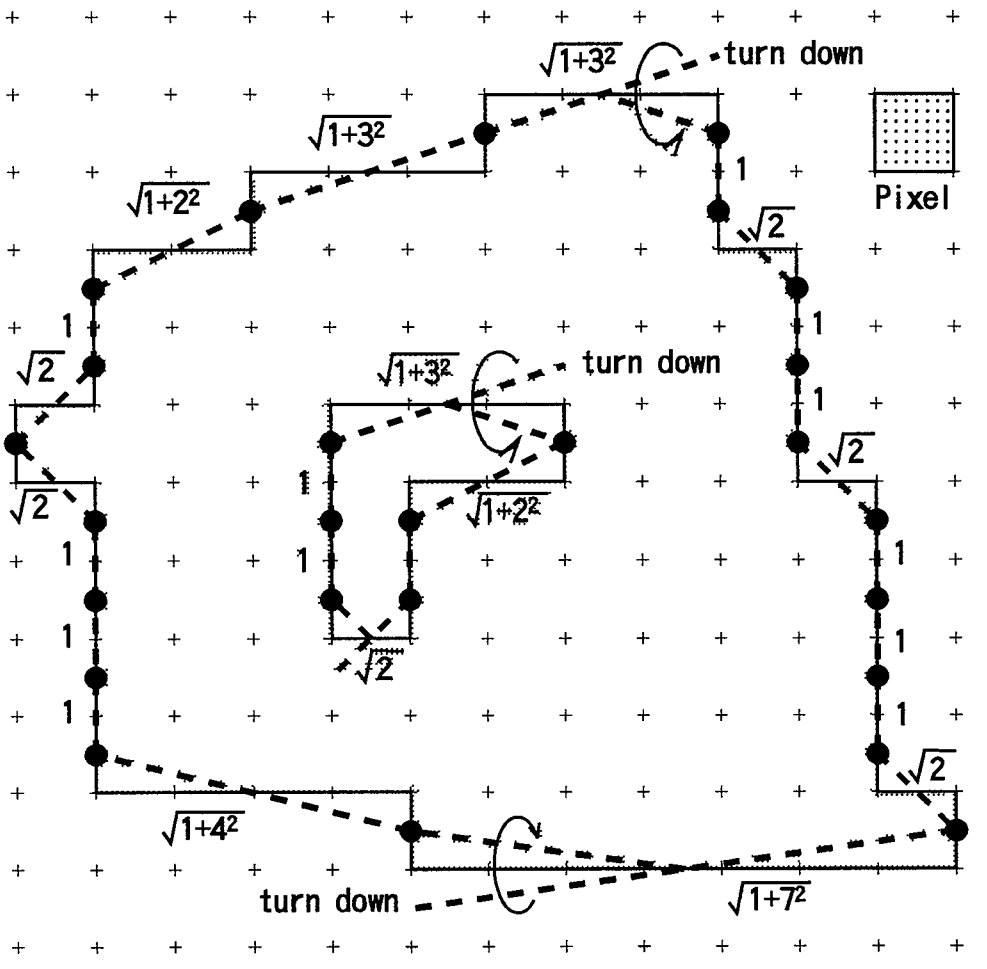


Fig.3.6 Example of perimeter calculation

今後このように表す． n は整数である．

図３．４及び図３．６から視察により

(a) 直線の傾きが $1:n$ の場合，すなわち傾き角 $\theta \leqq 45°$ のとき，その長さは $\sqrt{1+n^2}$ と評価され真の長さと一致する．（ $n \geqq 1$ ）

また

(b) 直線の傾きが $n:1$ の場合，すなわち 傾き角 $\theta > 45°$ のとき，その長さは $n-1+\sqrt{2}$ と評価される．（ $n > 1$ ）

ただし $\theta > 90°$ については Y軸対称に折り返して考える．

３．２．４ 傾きが整数比の場合の直線エッジの長さ評価

２章 定理(1),(2)により直線エッジが２値化された時どのようなステップを生じるかを明らかにした．個々のステップの長さ評価は３．２．３項(a)，(b)に示す通りであるから傾き b ： a の直線エッジの ＤＬＳ 法によって求まる長さ l_mは

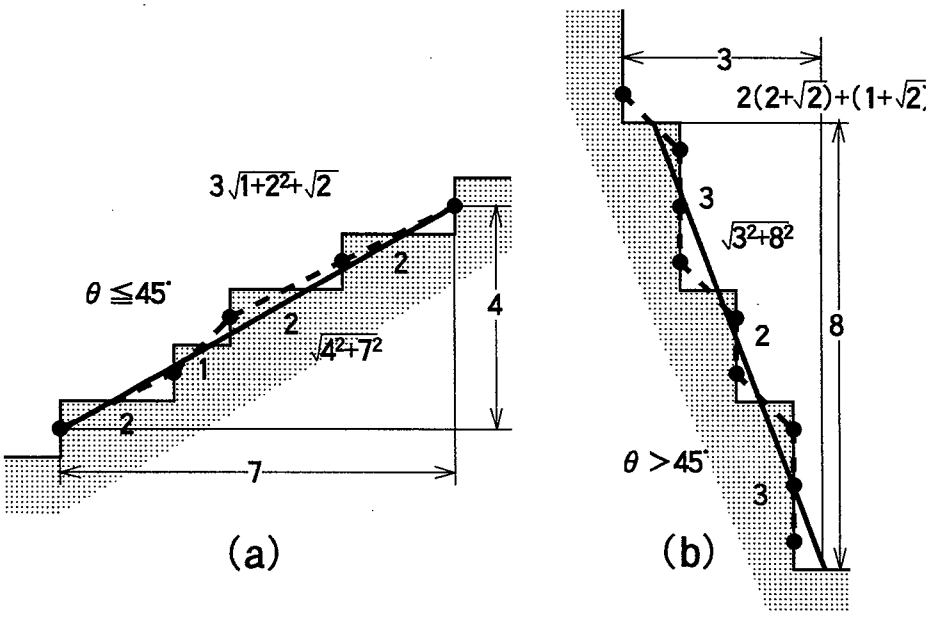


Fig.3.7 Measured length by DLS method

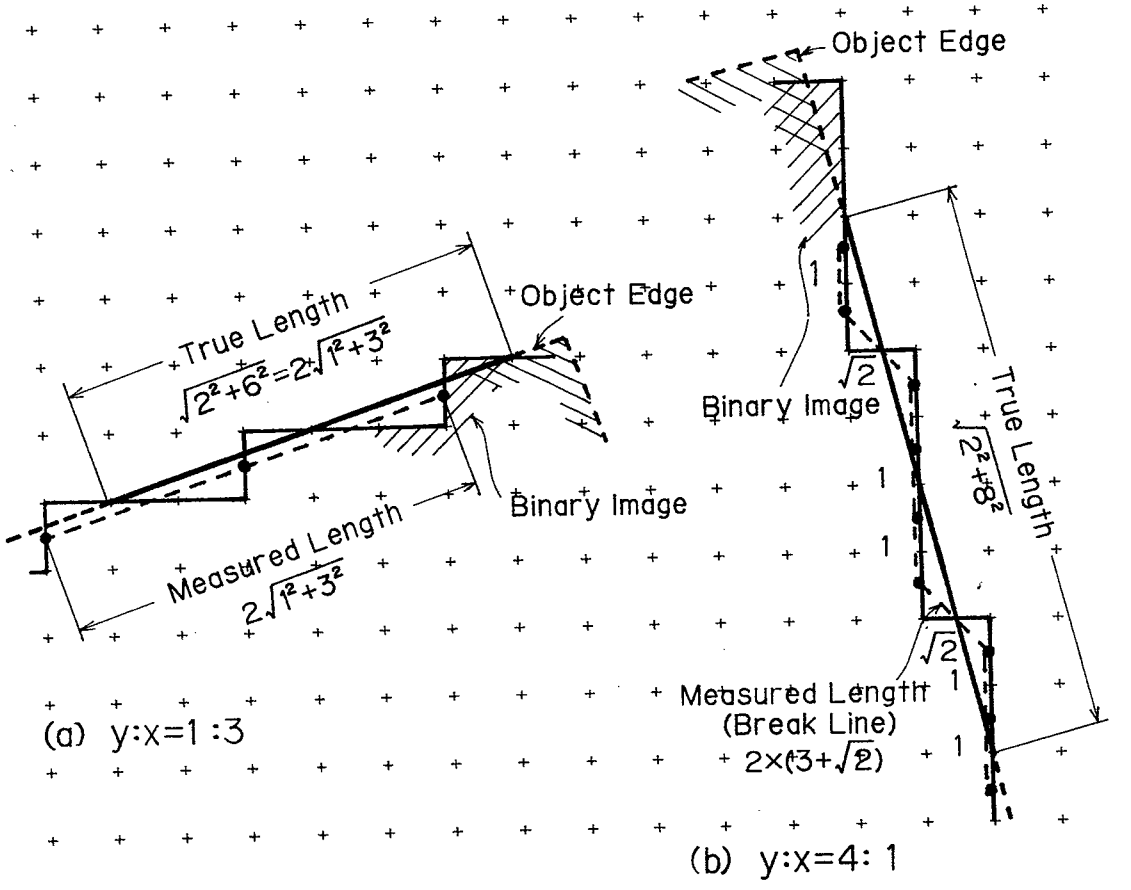


Fig3.8 Measured length by DLS method (slope -- 1:n and n:1)

$b \leqq a$ のとき ($\theta = \tan^{-1}(b/a) \leqq 45°$)

$$l_m = p\sqrt{1+n^2} + q\sqrt{1+(n+1)^2} \qquad \text{---(3.1)}$$

$b > a$ のとき（$\theta = tan^{-1}(b/a) > 45°$）

$$l_m = p\{n-1+\sqrt{2}\} + q(n+\sqrt{2}) \qquad ---(3.2)$$

真の長さl_tは　　$l_t = \sqrt{b^2 + a^2}$

この様子を 図３．７ に示す．破線がＤＬＳ法によって測定される長さ，実線が真の長さである．

図３．８は直線の傾き比の一方に１を含む場合の測長結果である．1:n の傾きの場合は破線は実線に沿って行くが n:1 の場合はジグザグが大きくなり，測定値は正の誤差を示すことがわかる．左図（ａ）は y:x=1:3 である．この場合は３．２．３項で述べた如く，測長直線は元図形のエッジと平行になり，真の長さと完全に一致する．これに対して右図（ｂ）は y:x=4:1 で図３．７と同じジグザグ現象が発生し，ＤＬＳ法による測長値は真の長さより長くなる．このように直線の傾きが 45° を境にして，測長精度が大きく変わることがわかる．図３．９はこの様子を示している．図では横軸に傾きθをとって縦軸に測長誤差 $(l_m - l_t)/l_t \times 100$ % を表している．45° ＜θ＜ 90° つまり $n-1+\sqrt{2}$ で評価される領域で誤差が非常に大きくなる．丁度 45° では図形エッジと測長線が平行となり誤差は ０である．

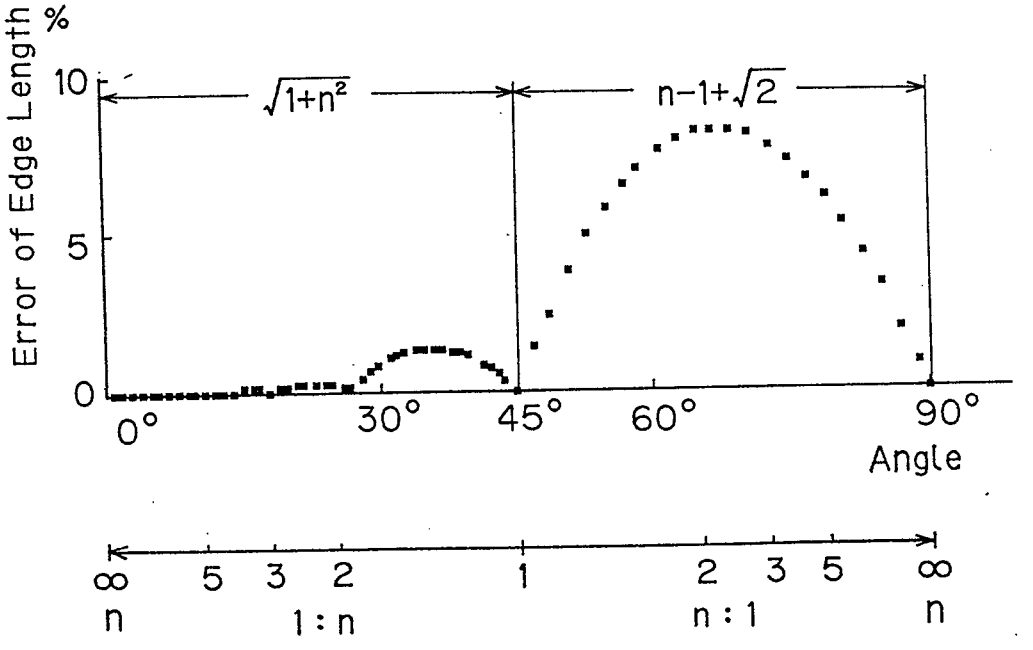


Fig.3.9　Error of measured edge length by DLS method

３．２．５　画素が長方形の場合の直線エッジの長さ評価

ラスタスキャン方式で画像を取り込む場合サンプリング周波数が変化すると物理的な画素寸法の縦横比が変わってくる．周波数が上がれば縦長になり下がれば横長になる．このような場合に DLS 法で評価される長さがどのように変化するかを考察する ．

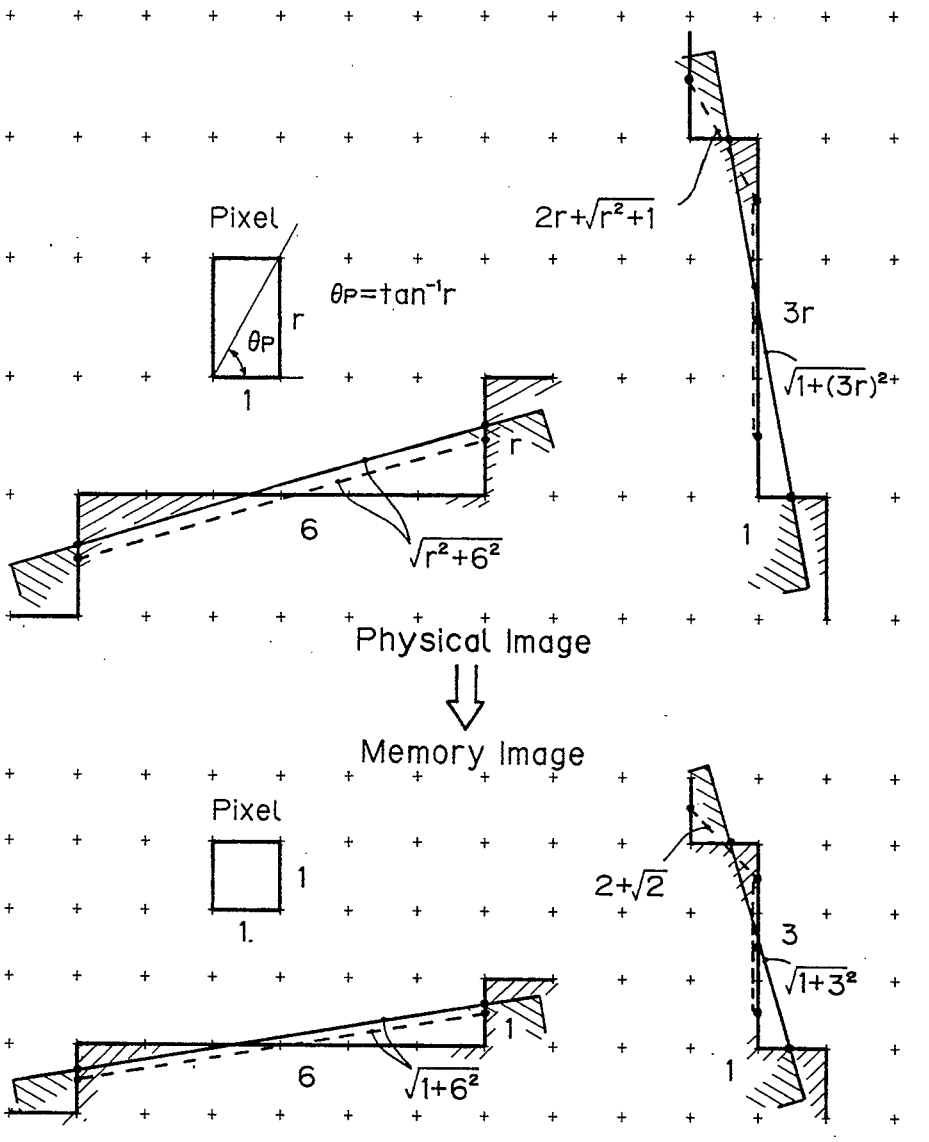


Fig.3.10 Physical image and memory image

図３．１０は画素の縦横比が $r:1$ の場合の物理的イメージとメモリイメージの関係を示している．ここで r は実数，以後縦横比を r で表す．また $\theta_P=tan^{-1}(r)$ を**画素角**と呼ぶ．傾き $r:n$ の直線エッジはメモリ内で $1:n$ になる．そこでメモリイメージから実際の寸法に換算するため ３．２．２項で示した ＤＬＳ 法の手順の中で $\sqrt{1+n^2}$ の計算を $\sqrt{r^2+n^2}$ に置き換える．この変更により３．２．３項(a)，(b)は以下のようになる．

(a') 直線の傾きが $r:n$ の場合，その長さは
$\sqrt{r^2+n^2}$と評価され真の長さと一致する．

(b') 直線の傾きが $nr:1$ の場合，その長さは
$(n-1)r+\sqrt{r^2+1}$と評価される．

ただし $\theta>90°$ については Y軸で折り返して考える．

物理イメージで $br:a$ の直線エッジはメモリ内で $b:a$ になるので２．３．１項（１）で述べた定理(1)，定理(2)は次の定理(3),定理(4)となる．但し画素の縦横比を r と仮定している．

定理（３）　傾き $br:a$（$b \leqq a$；正の整数，$\theta \leqq \theta_p$，r；実数）の直線エッジの一部が２値化されたときのイメージは　$1:n$　と　$1:n+1$　の２種類のステップで構成される．但し n は a を b で除したときの整数部である．それぞれのステップの個数 p, q は

$$p=(n+1)b - a$$

$$q=a-nb$$

定理（４）　傾き $br:a$（$b > a$；正の整数，$\theta > \theta_p$，r；実数）の直線エッジの一部が２値化されたときのイメージは $n:1$ と $n+1:1$ の２種類のステップで構成される．但し n は b を a で除したときの整数部である．それぞれのステップの個数 p, q は

$$p=(n+1)a-b$$

$$q=b-na$$

(a')，(b')，定理(3),(4)より画素の縦横比が $r:1$ のとき傾き $br:a$ の直線エッジの長さ l_r は　ＤＬＳ　法によって次のように評価される．

$\theta = \tan^{-1}(br/a) \leqq \theta_p$　のとき

$$l_r = p\sqrt{r^2+n^2} + q\sqrt{r^2+(n+1)^2} \qquad \text{---(3.3)}$$

$\theta = \tan^{-1}(br/a) > \theta_p$　のとき

$$l_r = p\{(n-1)r+\sqrt{r^2+1}\}+q\{nr+\sqrt{r^2+1}\} \qquad \text{---(3.4)}$$

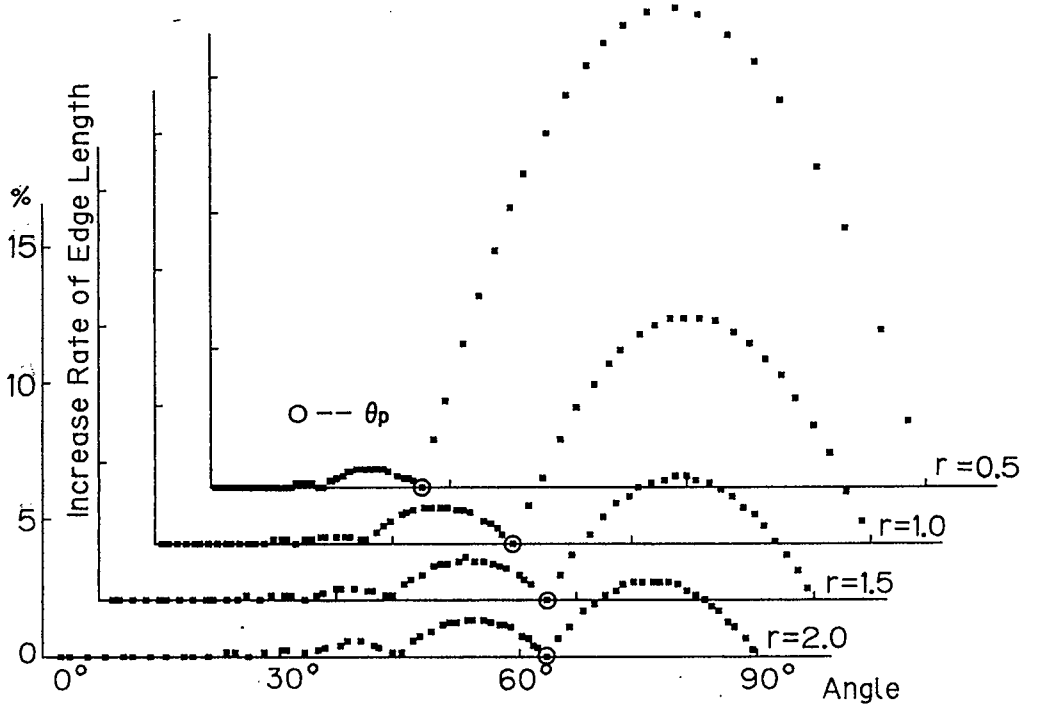


Fig.3.11　Increase rate of measured edge length

ただし真の長さ　l_tは

$$l_t=\sqrt{b^2r^2+a^2}$$

図３．１１に r が変化したときの直線エッジの傾きθとエッジ長測定誤差の関係を示す．r を大きくとることにより誤差はθが画素角θ_pより大きい領域で急激に減少する．丁度画素角に等しい時，誤差は 0 である．画素角より小さい領域では r=1.5 近くで一度増加するがその後再び減少傾向となる．r をさらに増加した場合全領域で単調減少する．このとき画素角以下では小さなピークを生じるようになる．画素角以上での最大誤差が画素角以下のそれより大きい傾向は変わらない．

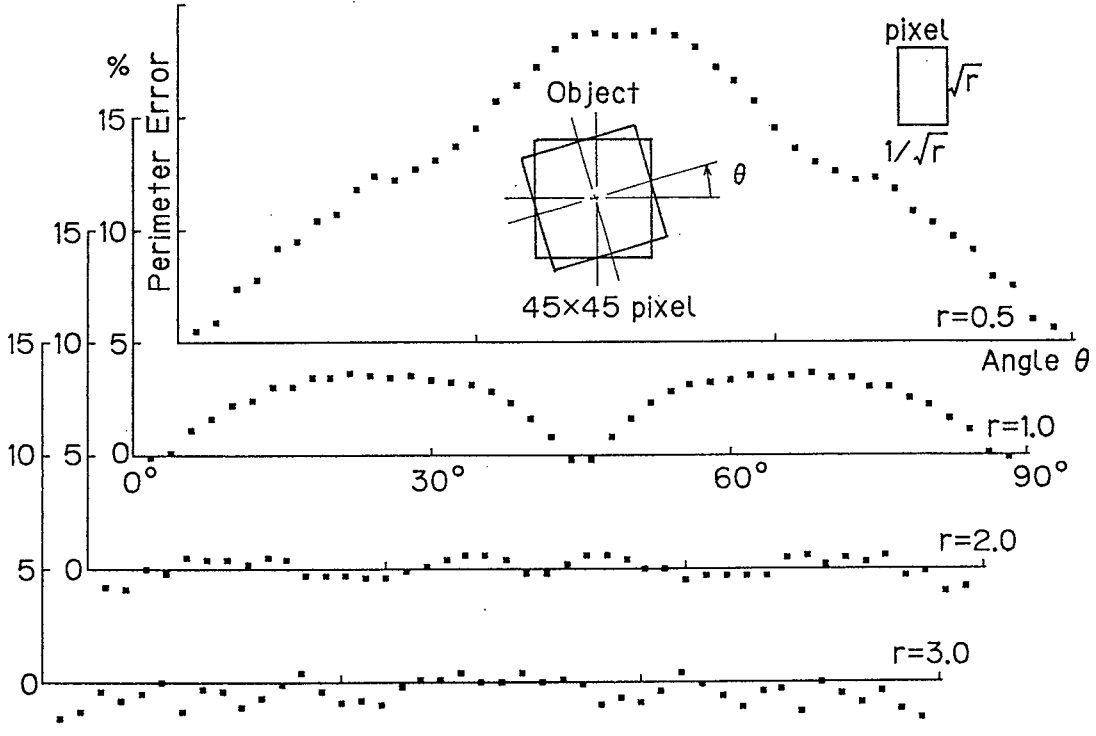


Fig.3.12　Measured perimeter for square (Area of pixel is constant)

３．２．６　コンピュータシミュレーション

現実の図形に対する本手順の測長特性をつかむためにシミュレーションを行った．これまで直線輪郭に対する吟味を行ってきたので直線部を多く含む図形として正方形を取り上げた．手順は次のとうりである．

(1)対象図形として正方形をとり，２値化メモリイメージを作成した．この図形を 0～90°の範囲で，３°きざみで，中心をシフト（９種類－－X, Y 方向　それぞれ 0，0.2，0.4 Pixel）しながら回転させ２値化を行った．２値化は図形が画素の面積の 50[%]以上を覆っている時を”１”とした．

(2)このようにして求めたメモリイメージに対してＤＬＳ法を適用して周囲長を求めた．ある角度に於ける９種類の周囲長データの平均値，及び標準偏差σ（バラツキ）を求め

た.

(3)画素の縦横比 r を変化させて(1),(2)を繰り返した.

図３．１２，図３．１３ はこの結果である．縦軸は真の長さからの誤差を示している．各データポイントは平均値を示している．図３．１２ は画素面積を一定（面分解能を落とさずに）にして r を変化させたものである．r が増加するにしたがって誤差が減少する様子がわかる．r=1のときどちらも角度４５° で誤差が０になっているこれは正方形の各辺が丁度傾き1:1の角度になり累積部分長の値がそれらの辺の長さと丁度等しくなるからである．逆にr=0.5では誤差が最も大きくなっている．これは先とは逆に累積部分長の評価長が最もジグザグに接続されるからである．

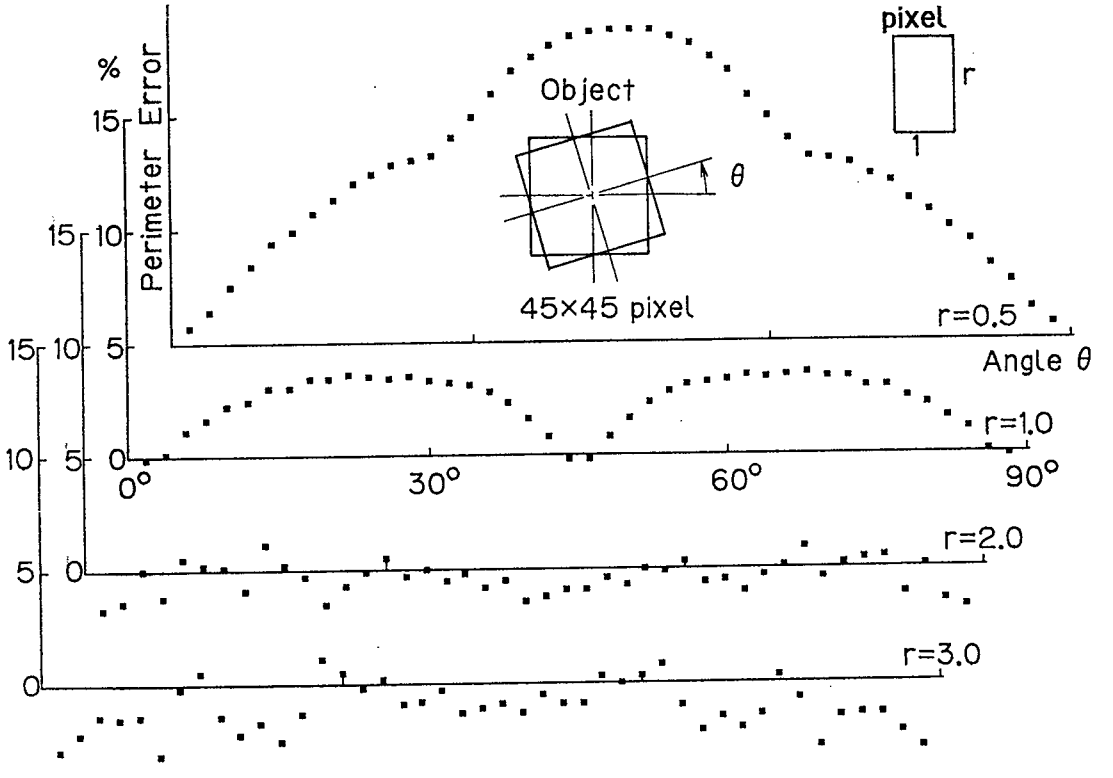


Fig.3.13 Measured perimeter for square (Area of pixel is not constant)

しかし r が 3 以上になると ①誤差が全体として負の値を示すようになる． ②誤差の絶対値が増える．従って r=2 の前後に最適値が存在すると考えられる．誤差は r=2 のとき最大 0.5 %前後であり多くの応用が期待できる．

図３．１３ は大きさを縦方向にだけ r 倍した画素を用いた場合である．図３．１２ に比べるとばらつきは大きいが同様の傾向を示している．誤差は 図３．１２ に比して若干増えて r=2 で 最大 0.9 %前後である．

表３．１は各データポイントで得られた標準偏差(0～90° 29個）の平均値を示している． r の増加と共に面分解能が低下する方（下段）は，ばらつきも増加している．面積分解能が変化しない方（上段）はばらつきの増加も僅かである．

３．２．７ 考察

Table 3.1 Mean value of standard deviation for different "r"

pixel \ r	0.5	1.0	2.0	3.0
$\sqrt{r}$ / $1/\sqrt{r}$	0.41 %	0.41 %	0.43 %	0.47 %
r / 1	0.34 %	0.41 %	0.52 %	0.57 %

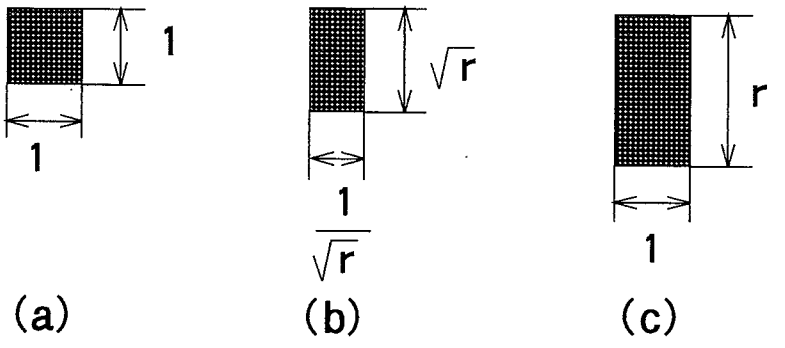


Fig.3.14 Form of pixel, physical size comparison

（１）縦横比 r の増大による測長精度向上の効果

縦横比rを1より大きくしたときの測長精度向上の効果は図３．１１からも明らかである．図３．１２（面分解能一定）と図３．１３（面分解能はrに反比例して低下）を比較すると図３．１３では画素の大きさが縦方向に伸びて大きくなり面分解能が落ちているにもかかわらず r の増大により誤差が減少している．図３．１４は画素の形の変更を実際の面積比で示している．（ａ）が基本的な正方形画素，（ｂ）が面積一定で縦長にしたもの（ｃ）は縦方向にr倍したもので面分解能は$1/r$となっている．

縦分解能を落としても良い結果が得られるのは分解能に起因する誤差の全体に占める割合がアルゴリズムに起因する誤差に比して小さいからである．分解能は回路技術上の問題であって比較的簡単に解決可能である．アルゴリズム誤差を減少させることがより重要である．

（２）縦横比 r の最適値

r の増加は DLS 法による周囲長測定について好結果をもたらす．しかし無制限に増加することは分解能の縦横バランスをくずす．つまり縦エッジに於ける左右方向の凹凸を見失う．図３．１２，及び図３．１３に於て r=3 以上で負の誤差が増えるのは正方形の頂点が平均化によって丸められ角がとれてしまうからである．したがって r の値は応用の用途により最適値がある．つまり鋭利な角を多く含む応用ではいたずらにrを大きくする

ことは得策ではない．逆に自然界に存在する一般的な物体を対象とするような場合にはrをわりと大きく採ることにより精度の向上が期待できる．縦横比rの値としては現在の所r =2　前後に最適値が存在すると考えている．

３．２．８ まとめ

本手順に従えば，縦長の画素を採用することによって周囲長，形状係数を高速，高精度で計測可能なことを理論とシミュレーションによって確認できた．１方向からの１スキャンで結果が出るので移動中の物体を連続して実時間計測するのに適している．

本節では特に直線エッジの問題に注目して解析しこれについて良好な結果を得た．図形が主として曲線で構成される場合には，これを円弧で近似することによって２値化の過程を分析し，DLS 法によって得られる長さがどのようになるかを解析できる．またこの手順を実行するハードウエアを製作して各種図形に対する測長特性を評価している．これについて３．３節で詳細を記す．

３．３　隣接２線走査ディジタル測長法のハードウエア化とその性能評価

３．３．１　あらまし

隣接２線走査法（ＤＬＳ法）の原理[7]については３．２節で述べた．その中でＤＬＳ法が，２値化された図形データの２点間のディジタル測長及び，周囲長，形状係数を高速に求める際の有効な手段であることを示した．そしてこの方法による測定精度が，縦長画素の採用によってより高められることを理論的に予測し，シミュレーションによってこれを確認した．また精度向上の理由について定性的考察を加えた．これは第２章でも予測していた．

これらの知見を基に画像入力，２値化を含めた実際の装置に於て，ＤＬＳ法の特性を総合的に検証すべく，この手順を実行するハードウエアを作製した．この装置を用い，直線，円弧を含む各種の図形についてその測長特性を検討したところ，シミュレーションによって得られた結果とよく一致した．エッジが円弧の場合について特に好結果を得たが，これは周囲が曲線で構成される一般的な図形に対して精度が高いことを示している．測定に要する時間は 1/60 [s]と高速である．回路構成もシンプルにでき，本手順の特長であるハードウエア化の容易さを確認できた．

ラスタースキャン方式で縦長画素を実現するには，横分解能を上げる方法と，縦分解能を落とす方法がある．ＤＬＳ法によれば，総合分解能が落ちると考えられる後者の方法でも，周囲長測定誤差が減少することをハードウエア上で実験的に確認した．実際には水平走査線を 1/2, 1/4 に間引きして縦長画素を実現した．

以上の検討の結果ＤＬＳ法が周囲長測定の方法として実用的に有効な手法であることを実証することができた．

３．３．２　ハードウエア構成

（１）　概要

理論及びシミュレーション結果を検証するため，ＤＬＳ法を実行するハードウェアを作製した．図３．１５ に全体のブロック図を示す．基本的にはベルトコンベア上の物体を 2048 画素の１次元イメージセンサでとらえて処理することを考えて設計しているのでア

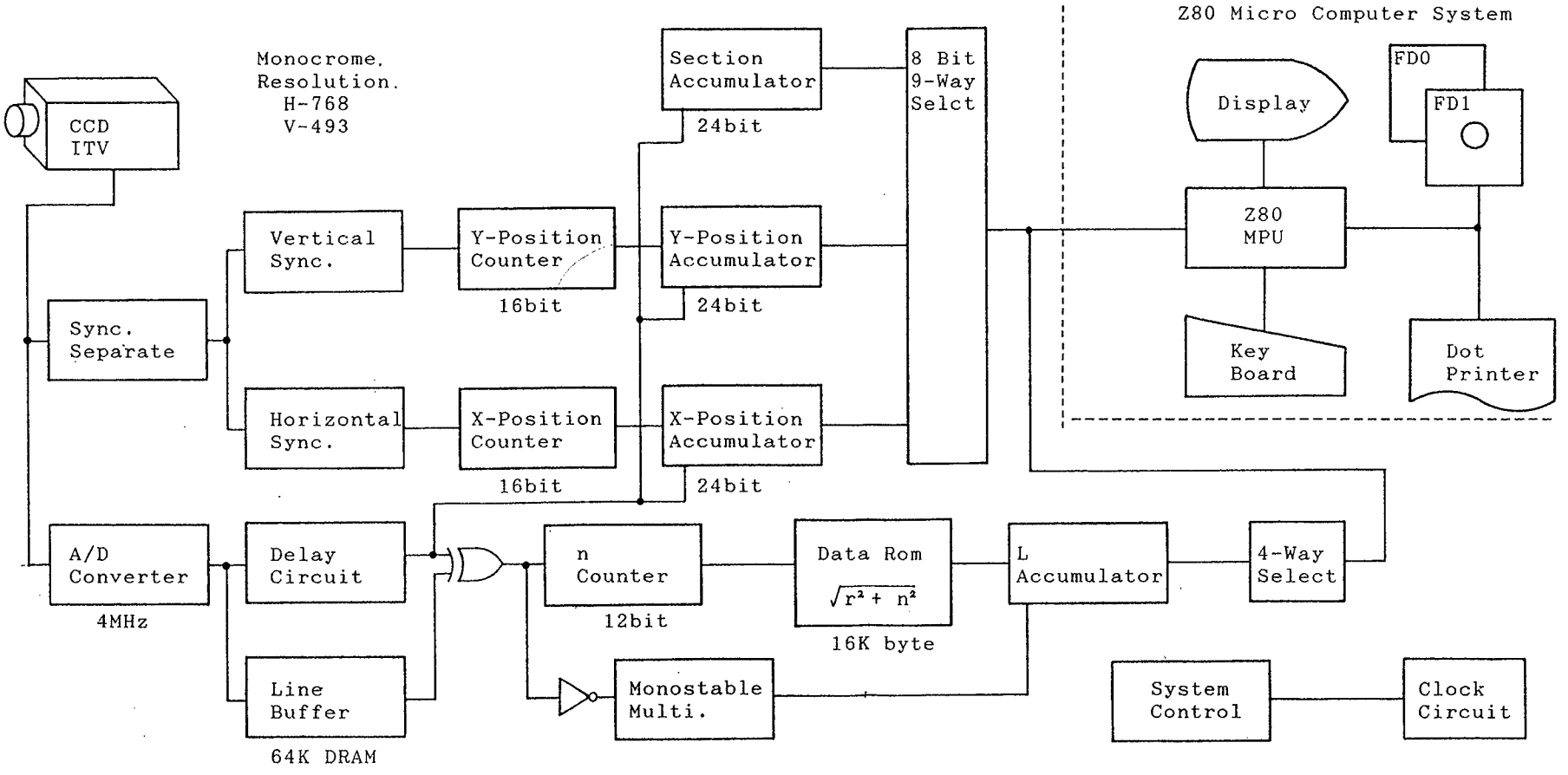


Fig.3.15 Block diagram of hardware

Table 3.2 Major specification of experimental device

Pick-up Device	SONY XC-77 CCD Camera
Dimension of Field	$350_H \times 240_U$ [mm]
Dimension of Pixel	n : 1.692 [mm] n = 1,2,4 r = 0.591 , 1.182 , 2.364
Total Pixels	5×10^4 [Pixel]
Distance Between Camera and Object	600 [mm]
Processing Cycle Time	279 [ns]/Pixel
Measured Data Output	During Vertical Flyback Period
Processing Time for One Field	1 / 60 [s]
Horizontal Resolution	204 Sampling Point
Vertical Resolution	240 Sampling Point (max)

キュームレータのビット長は比較的大きくとっている．(24bit)

撮像装置としては工業計測用カメラを接続した．図中，中央付近にある４個のアキュームレータの出力は２４bitのバスによりコントローラと結ばれている．コントローラはこれらのデータの取り出しと重心，周囲長に関する乗除算を行い，結果をディスプレイに表示する．また上位装置とのインターフェース機能を持つ．この構成で周囲長のほか面積，重心（X, Y），形状係数を求めることができる．$\sqrt{r^2+n^2}$ の計算は ROMを使用したテーブルルックアップ方式によった． 形状係数Fの定義式[2]は１．２．２項で定義した次式を採用した．

$$F=L^2/(4\pi S) \qquad \text{---(3.5)}$$

L；周囲長　　S；面積

式(3.5)の計算はコントローラで行う．周囲長測定値の縦横比依存性の検討のため画素の縦横比 r を変えることができるようにした．ここで画素の縦横比 r とは画素の縦寸法を横寸法で除したものである．シミュレーションの結果から r= 2 前後を想定し，実際には他の制約から r=0.591, 1.182, 2.364 とした．サンプリング速度を一定として，水平走査線を２回に１回，或は４回に３回抜き取る方法によった．したがって面分解能はr に反比例して低下する．これは３．２．６項で述べたシミュレーションの条件に一致する．今回制作した実験装置の主要諸元を表３．２に示す．イメージセンサは既製の計測用モノクロ CCD カメラを利用した．

（２）ＤＬＳアルゴリズムの実行部

ＤＬＳ法では常に，隣接する２本の走査線上の４個のデータ(h, i, j, k 図３．１６ 参照)をアクセスしながらシーケンシャルに計算を進める．リアルタイム処理においては i, j は時系列に入力される生のデータ，h, k は一走査線分の遅延バッファにより得られる．j, k はそれぞれi, h の１ Pixel 分の遅延データとして得られる．

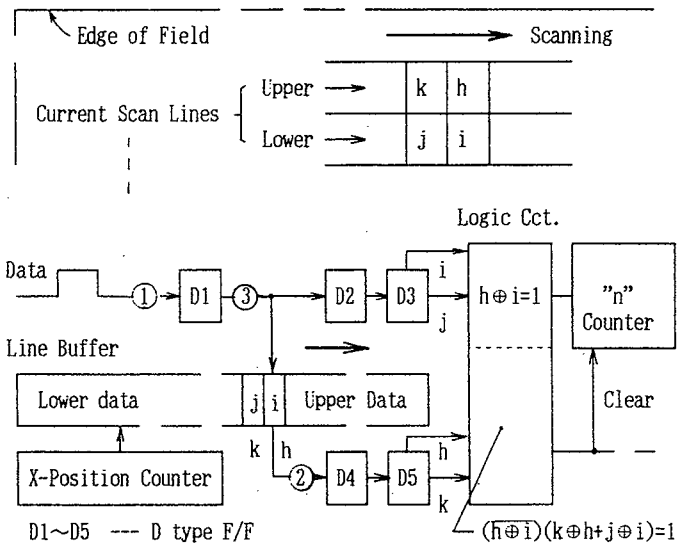


Fig.3.16 Core block executing DLS algorithm

図の下部にこの様子を示す．ラインバッファのアドレス線には水平方向の位置を決めるxポジションカウンタの出力が入る．D1～D5は D 型フリップフロップである．図で処理サイクルを説明すると次のようになる．

① xポジションカウンタが＋１されてアドレスが確定したのち，その Pixel のデータが入力される．これをD1にラッチする．

② ラインバッファから１ライン前の当該アドレスのデータ h を読みだし D4 にラッチする．

③ 同じアドレスに D1 の出力を書き込み，同時に D2 にラッチする．

D3 は D2 から１サイクル遅れのラッチであるから D3 には j が保持されている．同様に D5 は k を保持している．h, i, j, k は組合せ回路に入り $h \oplus i$="1"，$(\overline{h \oplus i}) \cdot (k \oplus h + j \oplus i)$="1" が検出される．前者は nカウンタを＋１し，後者は ROM から$\sqrt{r^2+n^2}$の値を読みだし累算器に加えた後 nカウンタをクリヤする．このあと①に戻って処理を続ける．ラインバッファとしてはスタティックＲＡＭに比して安価なダイナミックＲＡＭを用いた．上述のごとく，周期的にアクセスするのでリフレッシュが不要だからである．

写真３．１に本装置の外観を示す．４枚のプリント板から構成され，それぞれはバスでホストコンピュータに接続されている．写真３．２はステップモータを用いた回転移動テーブルである．同じく写真３．３は直線移動テーブルである．これらは１つのホストコンピュータで管理され，データ採取の自動化がなされている．

３．３．３ ハードウエアの評価

回路のデバッグのため表示色 ３ チャンネルの専用ロジックアナライザを製作した．これを用いることによってディスプレイ上で図形の２値化後のパターンや，$\sqrt{r^2+n^2}$ を計算するタイミングが目視確認できる．静止した図形を撮影し，アナライザ上で２値化の結果に変化がなければ得られる測定値も一定である．つまり形状データが一定なら得られる結果は手順により一意的であるというアルゴリズムに起因する誤差の性質が確認できる．

Photo.3.1 Experimental Device

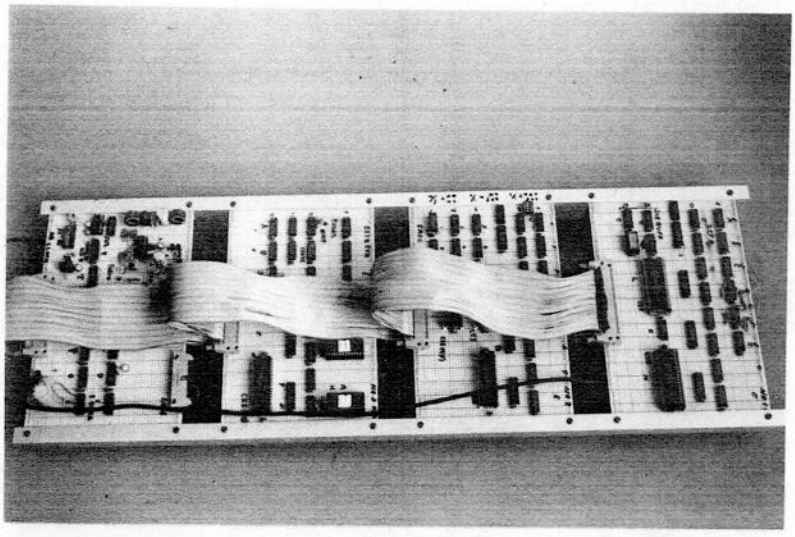


Photo.3.2 Rotating table

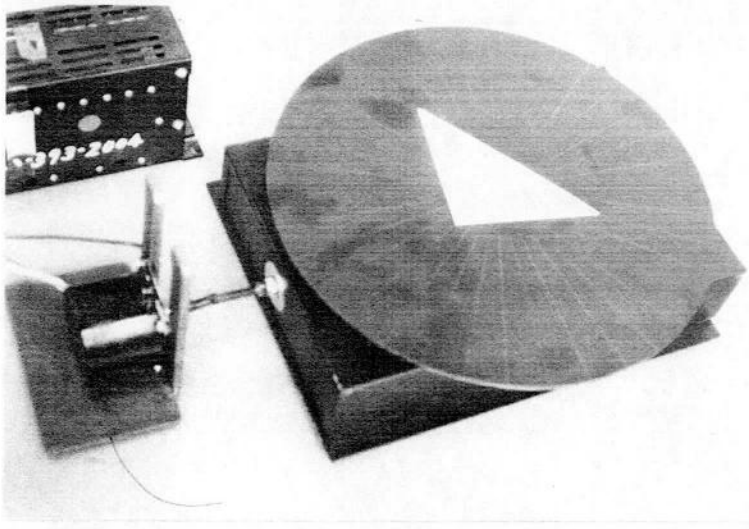
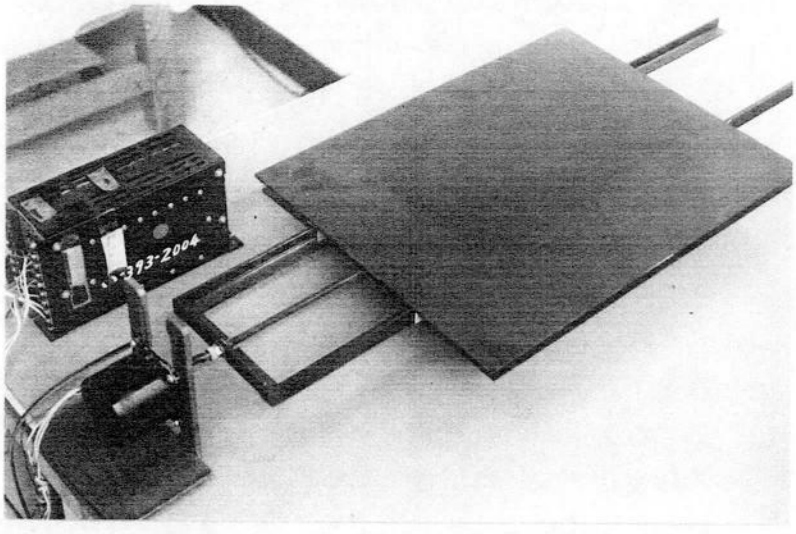


Photo.3.3 Lincar positioning table



（２．３．３項参照）

（１）面積の測定

形状係数は周囲長と面積から計算されるので，面積の測定精度は形状係数の測定精度に影響を与える．面積に関して 100 × 100 mm（5910 Pixel）の正方形を水平から　2°ずつ９０°まで回転移動した場合の，正方形の面積測定値のばらつきを図３．１７に示す．図中左側は度数分布を示す．縦軸は面積の真値からの相対誤差である．　r の値は画素縦寸法／画素横寸法である．　r の増加で画素面積が大きくなり，面分解能が低下するのでばらつき（標準偏差 σ）が大きくなっていることが判る．測定値の平均値はほとんど変化がない．r=0.591，1.182，2.364　のいずれの場合も 3σが0.6%以下で精度は非常に高い．

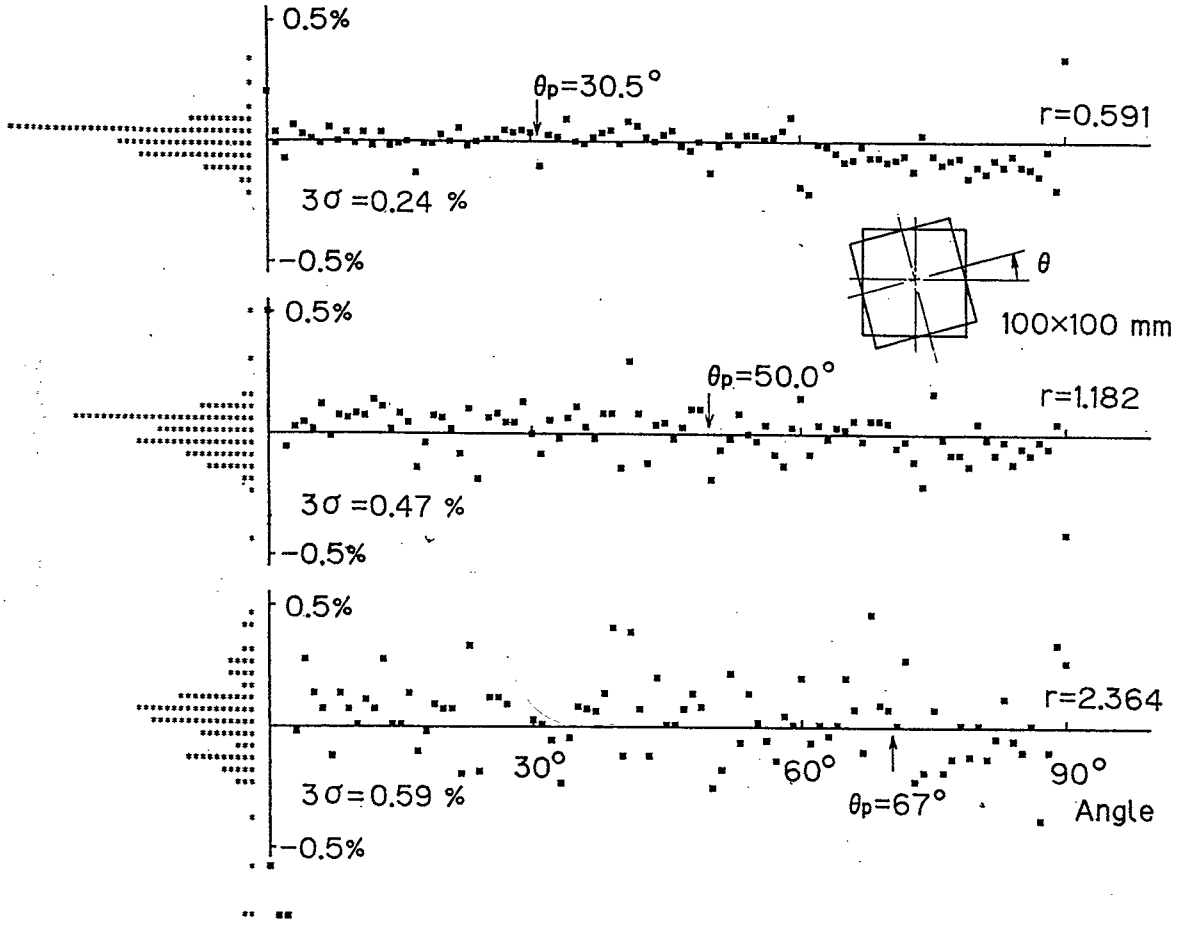


Fig.3.17　Relative error of area measurement (Rotating object)

図３．１８は同じ図形を 45° の傾きにおいて 0.05 mmきざみで直線移動させたものである．移動は約1.5 Pixel の範囲で行った．前図と同様に度数分布と，縦軸を描いている．Position 0 mm の位置で画素中心と正方形の中心を一致させている．この図では 3σは最大で0.19 %である．画素寸法の増加と共にばらつきが増す傾向は回転移動と変わらない．後に示すがこの面積測定精度は周囲長測定精度より，はるかに高いので，形状係数の精度は周囲長精度に依存することになる．画素の縦横比　r を増加させるとばらつきは増加するが比例値より少ない．これは分解能に起因するものと，これに無関係に一定の電気的，

光学的ノイズが複合してその原因になっているからである.

面積測定値は以下に示す周囲長測定の校正基準として用いた. すなわち周囲長測定値はカメラと被写体との距離, 光学系の倍率等により変化するので, これを校正するため100×100 [mm²]の正方形の面積を測定して5910[pixel]=(100×100)/(1×1.692)のデータが出る位置, 環境に装置を設定した.

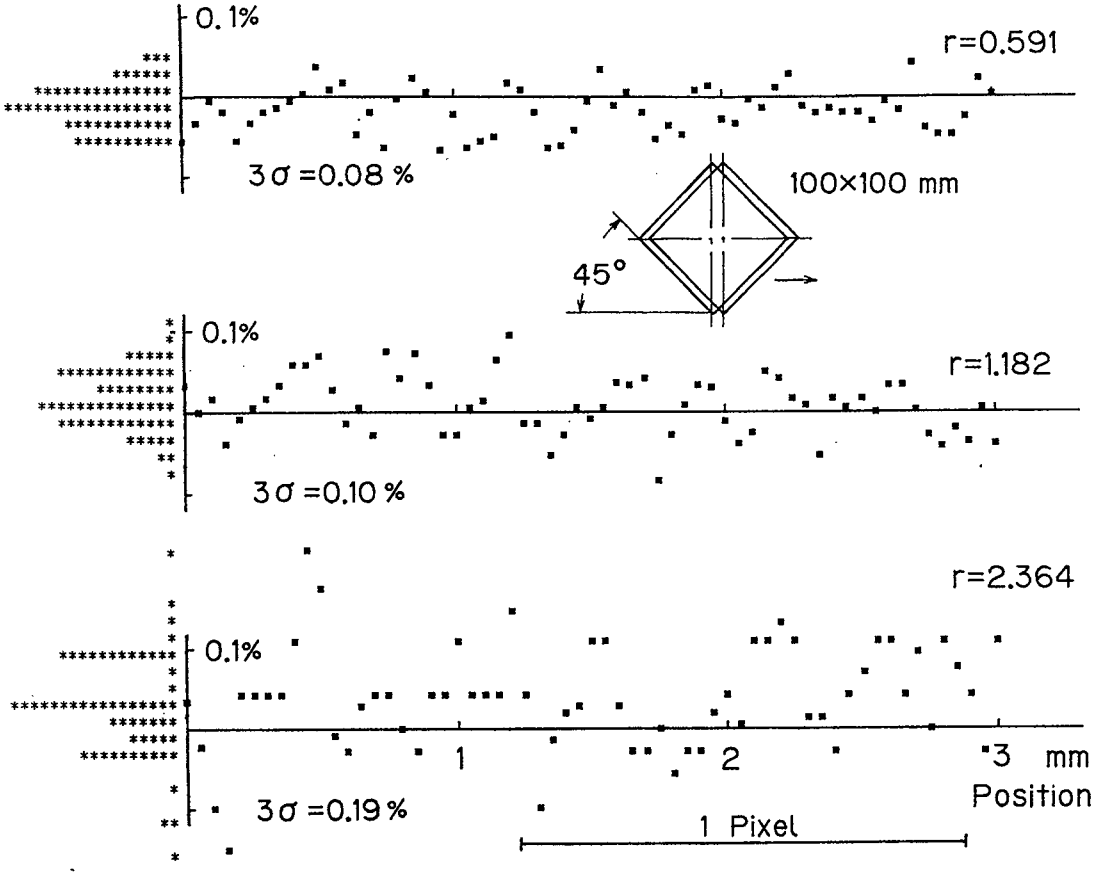


Fig.3.18 Relative error of area measurement (Linear movement)

(2) 直線エッジの長さ測定評価

直線エッジの長さ測定値がその傾きによってどのように変化するか実験を行った. 図形は図3.19中に示す長さ200mmの細片を用いた. r=0.591のとき巾2.0 mm, r=2.364のとき巾5.0 mmである. 短辺の割合は4/400=1 %及び10/400=2.5 %である. 巾が狭いほうが長片のエッジ長の影響がより大きく出るので理論解析の状態に近くなる. 巾をrに応じて変えたのは, 狭すぎると2値化の結果が"0"になるからである. 図3.19はrをパラメータとして, 細片の傾きを変えて周囲長を測定し, 真の値(r=0.591のとき404 mm, r=2.364のとき410 mm)を基準としてこれからの誤差を示している.

rの増加による誤差の改善の様子がわかる. 図3.20がこれに対応する理論解析[7](3.2.5項参照)の結果である. ほぼ理論値に近い結果が出ている. r=2.364のときの実験値の最大誤差は3 %であるが光学系の歪による視野内偏差や, 走査線をスキップし

て分解能が 1/4 に落ちていることを考慮すると理論値 2 %に十分近い値である.

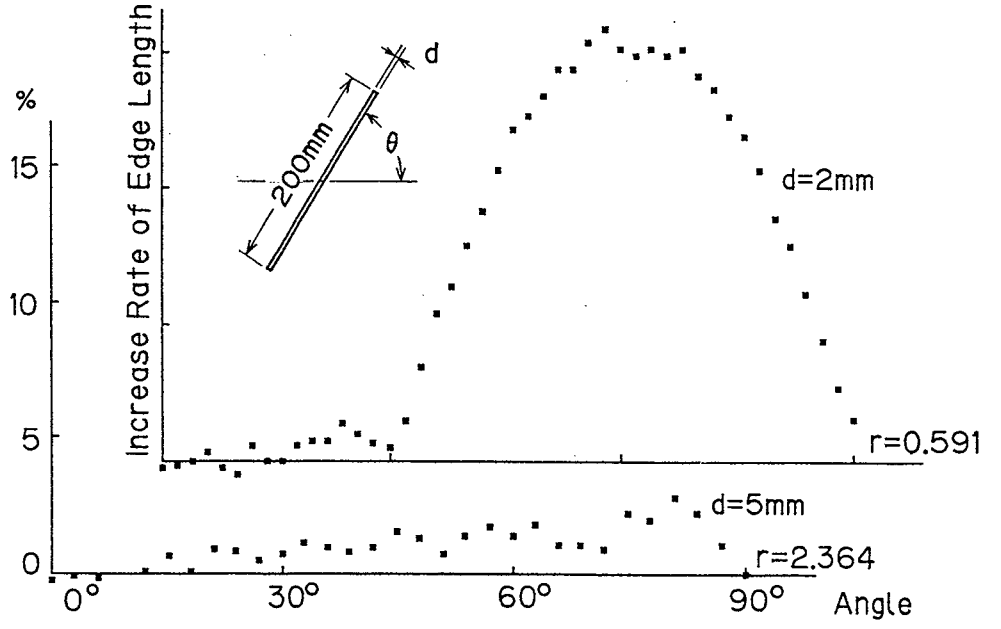


Fig.3.19 Increase rate of edge length measured by DLS hardware

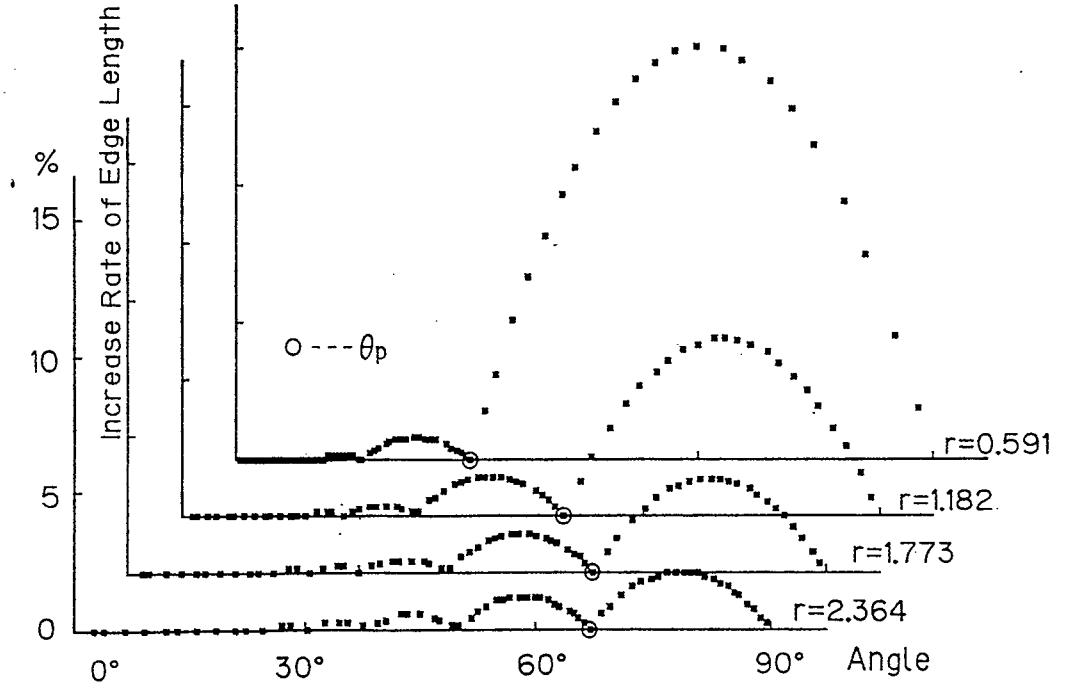


Fig3.20 Increase rate of edge length (Theoretical value)

（３）正方形の周囲長測定評価

角度の異なるいくつかの直線からなる図形では平均化され，特定の姿勢で（２）で述べたような大きな誤差を示すことがなくなる.

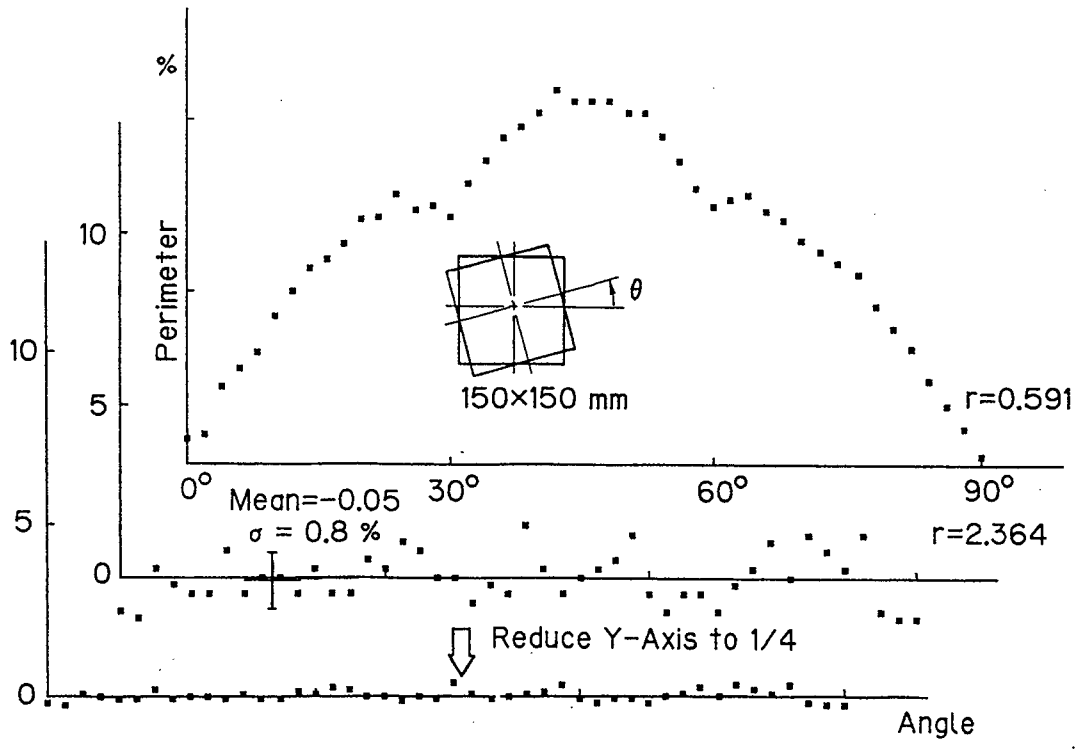


Fig.3.21 Relative error of perimeter for square figure measured by DLS hardware

図３．２１に正方形について，r の増加による改善の様子を示す．縦軸は一辺の長さ 150×4=600 [mm]を基準としてこれからの誤差を示している．図３．１３に示すシミュレーション結果（３．２．６項参照）と傾向が一致している． r=2.364 の場合σ= 0.8 %で精度も同程度である．同図最下段は先に述べた分解能の低下の影響を補正するためＹ軸を 1/4 に縮めて描いている．これより画素の面積を一定として r を変化させたときの精度が推定できる．σは 0.2 %である．

（４）直線，円で構成される図形の周囲長測定評価

いろいろな角度の直線及び円弧を含む一般的な図形の場合，同時に最大誤差が発生することはないのでアルゴリズムに起因する誤差の割合は（３）に述べた場合に比してさらに小さくなる．表３．３は，図３．２３ に示す，形状係数の理論計算が容易な各種の図形を対象として，周囲長と形状係数それぞれの理論値と実験値を示している． r=2.364 の場合である．図形の面積はいずれも 14400 mm^2= 2128 Pixel に統一している．データは図形を 0～90° の間を 2 ° きざみで回転させ 46 点採った．

これより平均値，理論値からの誤差，σを求めた．このデータより次のことがわかる．

① 尖点を多く含む図形は２値化により角が取れるため誤差が負となる． DLS 法によるエッジ長の評価は直線に対して正の誤差を示すことをみてきた[7]（３．２．４項参照）がこのこと以上にまるめの影響が大きい．

② 形状係数のσは周囲長のそれに対して約２倍である．これは形状係数が 長さの２乗の次元をもつ値の比として定義されるからである．形状係数のばらつきには面積も関係す

るが周囲長のばらつきの影響が大きい.

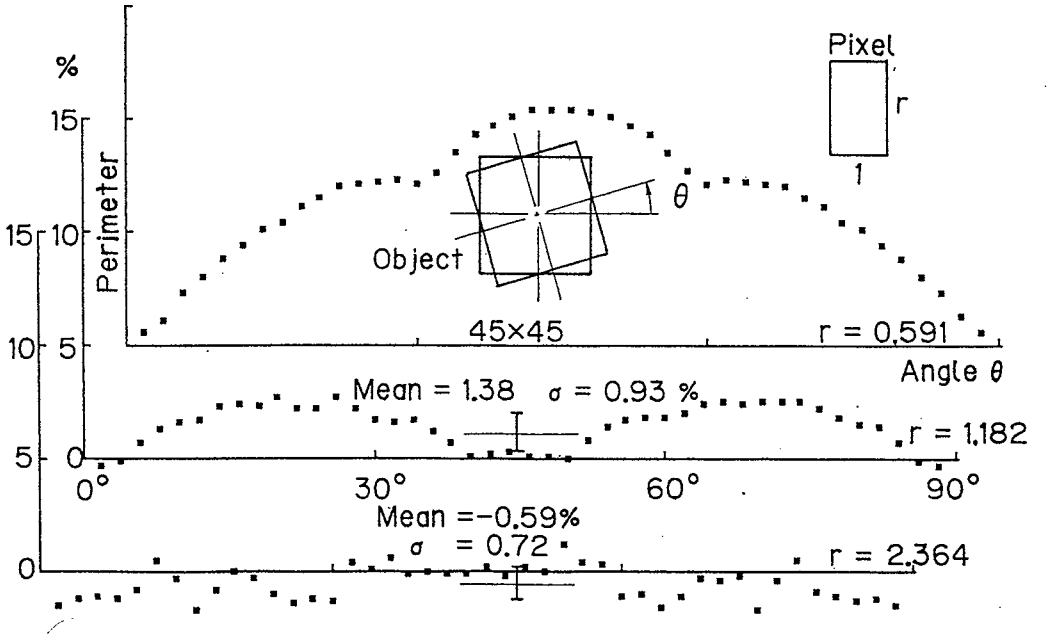


Fig.3.22 Relative error of perimeter for square figure (Result of simulation)

Table 3.3 Perimeter and Shape factor measured by DLS hardware for object figures in Fig.3.23

	NO.	L (Perimeter)				F (Shape Factor)			
		Ideal [Pixel]	Mean [Pixel]	Error [%]	σ [%]	Ideal [ ]	Mean [ ]	Error [%]	σ [%]
Composed	①	480	480	0	0.6	1.273	1.260	-1.02	1.3
by	②	720	717.0	-0.42	0.5	2.865	2.816	-1.71	1.1
Lines	③	960	950.7	-0.96	0.8	5.093	4.949	-2.83	1.1
	④	819	816.1	-0.35	0.8	3.707	3.622	-2.29	1.3
	⑤	1159	1146	-1.12	0.53	7.423	7.079	-4.63	1.2
Composed	⑥	424	431.1	+1.67	0.32	1.000	1.013	+1.30	0.33
by	⑦	603	612.4	+1.55	0.21	2.000	2.023	+1.15	0.47
Circles	⑧	851	857.1	+0.71	0.22	4.000	4.040	+1.00	0.37
	⑨	855	857.5	+0.35	0.18	4.035	4.056	+0.52	0.37
	⑩	927	928.1	+0.14	0.21	4.740	4.717	-0.46	0.33

③ 円で構成される図形では傾きが連続して変化するので，２章の例で見られるような，直線の２値化の際に発生する跳躍的な測定値の変化がなくばらつきは小さい．誤差は正であって後述，図３．２７ に示すシミュレーション結果と符合する．これは自由曲線で構

成される一般の図形について精度が高いことを示唆している．逆に直線で構成される図形に関してのばらつきを抑えるにはこの跳躍的な変化を少なくする工夫が必要であることを示唆している．（２．３．３項参照）

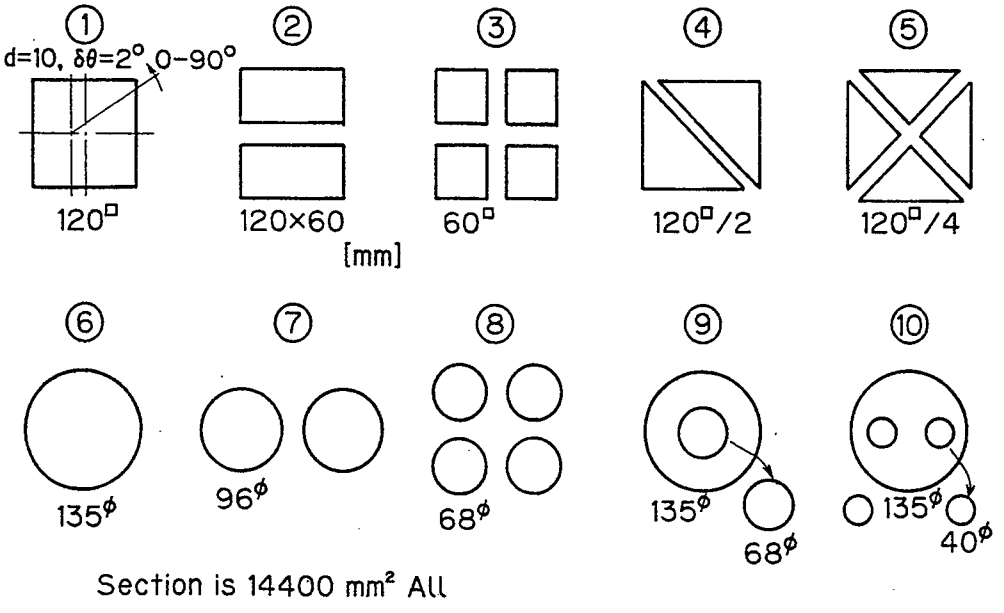


Fig.3.23　Object figures

（５）微小画素図形に対する評価

図３．２４に３種類の図形について，相似性を保って面積を縮小した場合の形状係数の測定結果を示す．r=2.364 の場合である．測定は一つの対象について 0～90° 間を 2° きざみで回転させ 46 点データを採った．これから平均値と理論値からの誤差，標準偏差を求め図中に示している．なお回転中心と図形の中心は 10mm オフセットした．一点鎖線は形状係数の理論値である．図形が任意の角度に置かれたとすると 99.7 %（3σ）以上の確率で判別できる面積は 10^2 Pixel 前後である．高速性を生かし角度を変えて測定し平均をとることを許すなら，20 Pixel まで判別可能である．この程度の図形は人の目でも区別がつき難い．

（６）光学系の歪みについて．

撮像素子としてのＣＣＤイメージセンサは完全な矩形の構造を持っている．しかし感光素子表面に写像される映像は光学レンズ系の不完全性のため，画像フィールドの中央部と端部ではその長さが異なって結像する．図３．２５はこの様子を示している．右図下は画像フィールド中央部における一水平走査線に対する映像信号である．右図上は画像フィールド上部におけるそれであり，中央部を100%とすると，δだけ長方形の寸法が縮んでいる．下端においても同じ現象が発生する．写真３．４はこの様子を示している．（ａ），（ｃ）つまり上端と下端では対象図形に対する長さが縮んでいるのが観測される 図３．２６は直径 30 mm の円を画像フィールド内の９箇所に置いて測定した結果である．

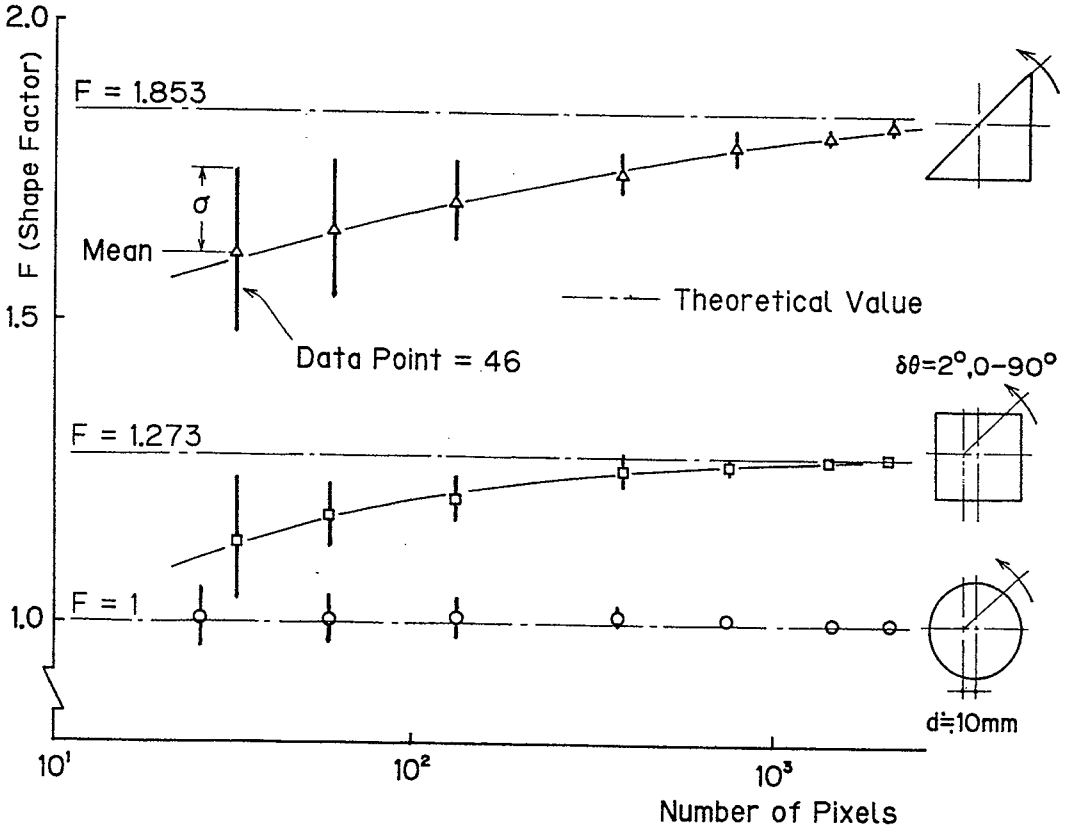


Fig.3.24 Measuring characteristics for small figures

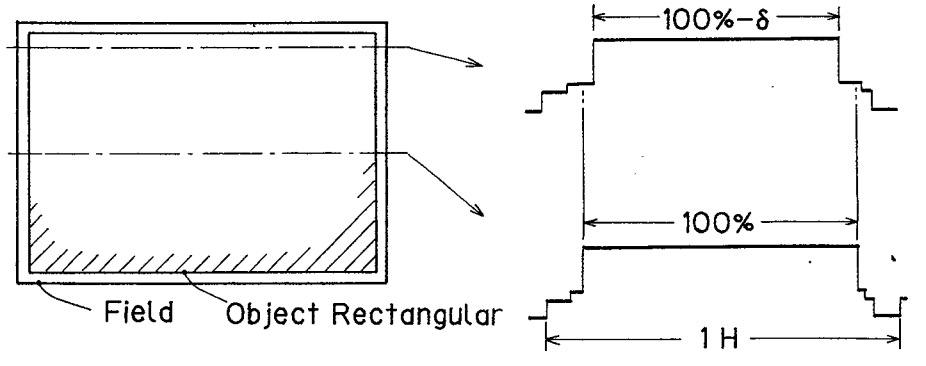


Fig.3.25 Shrinking effect on upper side of image field

左図は対象図形の配置と面積縮小の割合を示している．中央部を基準としている．右図は同様に中央部を基準として寸法縮小の割合の関係を示している．中央部を中心に上下左右対称に歪みが発生することが予想されたが実測ではそうはなっていない．これらの歪みは周囲長，面積測定値に直接影響を与える．本章に挙げた実験結果はこれらの歪みを包含して測定された結果であるから精度の高い光学系を用いれば，実験結果はもっと良い方向に改善される．

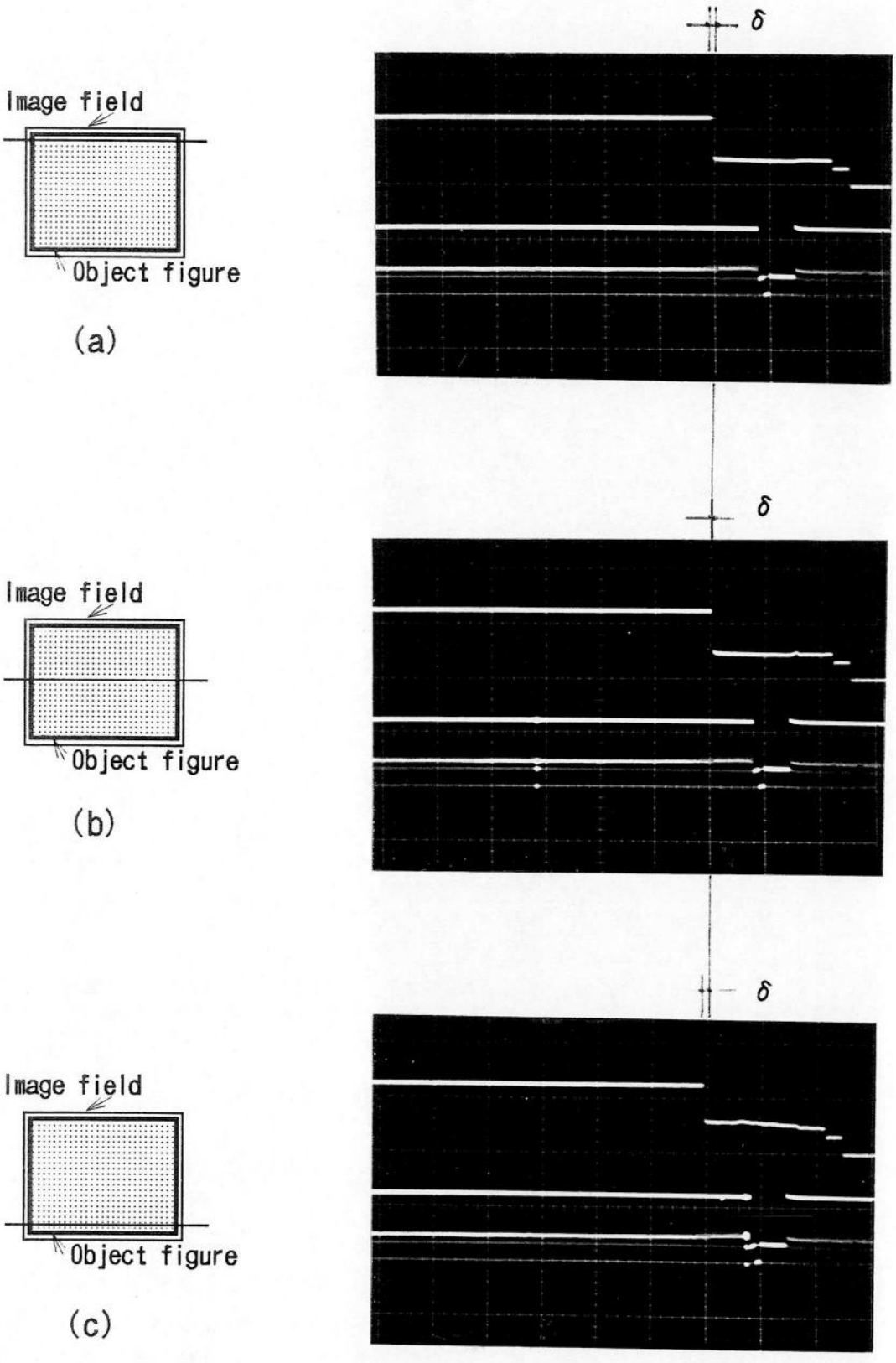


Photo.3.4 Optical distortion on edge of image field

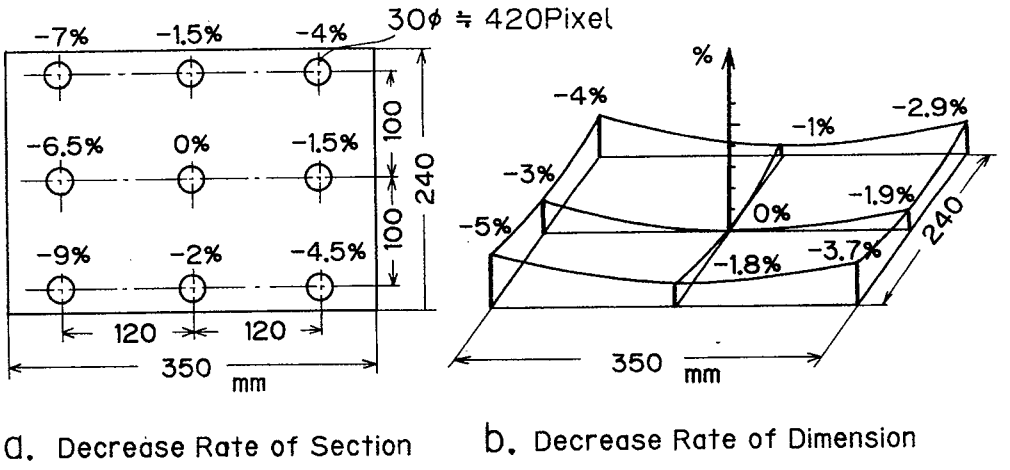


Fig. 3.26 Reduction of section and dimension depending on the optical distortion

3．3．4 考察

（1）他の測長法との比較

写像変換を用いない周囲長の測定法として ① 4連結画素計数法[2),(3)]，② 8連結画素計数法[2),(3)]，③ $\sqrt{2}$ 法[2),(3)]，④ $\sqrt{2}$-$\sqrt{5}$法[2),(3)]，⑤ 3画素ベクトル法[3)-(5)]などがある．精度はほぼこの順に高くなり手順の複雑さに比例する．直感的にはＤＬＳ法の精度は ④ $\sqrt{2}$-$\sqrt{5}$ 法に相当すると考えられる．なぜならＤＬＳ法の手順はX方向に④の拡張である$\sqrt{1+n^2}$ 法を，Y方向に$\sqrt{2}$法を適用し，Y方向の誤差の増加を縦長画素の採用により抑えたものと考えられるからである．3．3．3項（3）の結果から最大誤差を 3σ=2.4 %とみて文献(4)に記されている各種測長法のデータと比較するとこのことを確認できる．

処理速度の点ではＤＬＳ法は最も優れている．従来の方法のように図形の輪郭追跡の必要がなく，規則的な（規則的なことはハードウエア化が容易であることを意味する．）4個のメモリアドレスデータのアクセスと判断を1フィールドの画素の数だけ，繰り返せば1回の測定が済むからである．

このことは視野内に多くの図形がランダムに配置されているような場合にも1回のスキャンで総周囲長が求まり個々の図形を分離（ラベリング）して処理する必要がないことを意味している．また一方向からスキャンによるため対象となる物体を画像入力のため一時停止させる必要がなく，画像をフリーズして転送することも不要で，移動中の物体を連続して実時間計測可能である．図形の内部に空洞を包含する場合，この周囲も合わせて積算する．以上のことは他の手法にはないＤＬＳ法の特長である．

Hough変換[1)]による周囲長測定法は2値化の際の画素の欠落の問題を含めて取り扱うもので考え方としては本手法より優位にある．しかし計算量が非常に多く処理速度は遅い．また精度の面でも本手法に比較できる文献データは見あたらない．

（2）円に対するシミュレーションと画素の縦横比 r の最適値

3．2．6項では，シミュレーションにより，図形が正方形のときに測長誤差が r= 2

前後で0になり，さらに r が増えると0を通り越して負になることを示した．

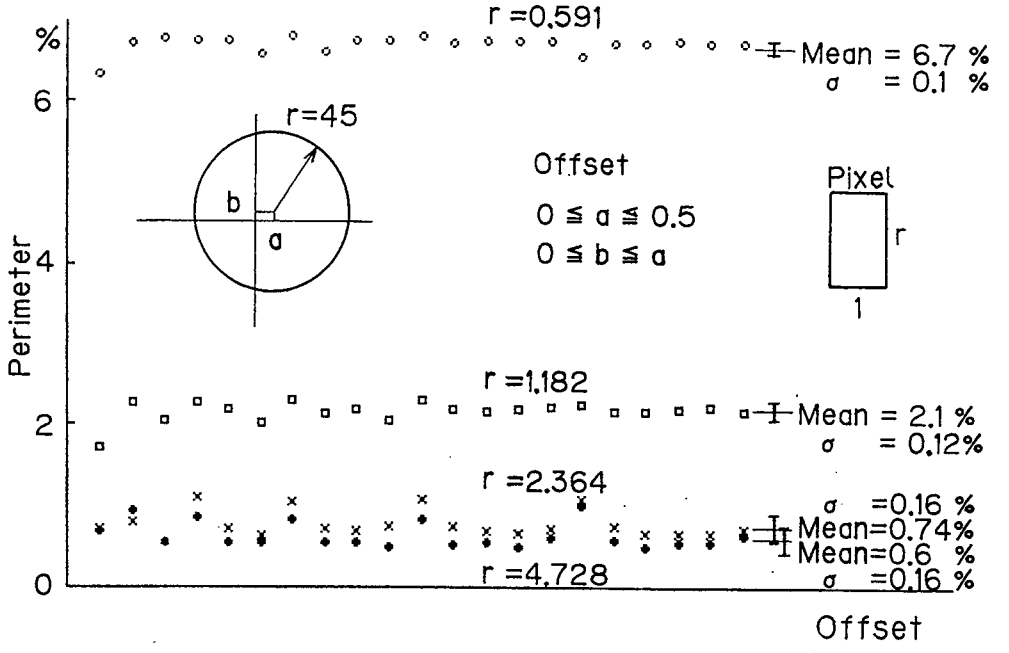


Fig.3.27 Relative error of perimeter for circular figure (Results of simulation)

これに対して図形が円の場合には 図３．２７ のような結果がシミュレーションによって得られている．図は半径 45 Pixelの円を 1 画素の範囲で中心をオフセットしながら，メモリ内に２値データを作製し，ＤＬＳ法により周囲長を計算したものである．a, b は 0.1 画素きざみで $0 \leqq a \leqq 0.5$, $0 \leqq b \leqq a$ の範囲で変化させた．バラツキの様子を示すため横に展開している．横軸は(a, b)=(0, 0), (0.1, 0), (0.1, 0.1), (0.2, 0), (0.2, 0.1), (0.2, 0.2) - - - に対応する．誤差は r の増加と共に0に漸近する．正方形のときのように負の領域に入ることはない．したがって直線や曲線が入り交じった一般的な図形の場合 r の最適値は実測データも考慮して2.0 ～ 3.0 が適当と考えられる．なお図３．２２と３．２７ を比較すると，バラツキは円の場合$\sigma = 0.16$ % であり，正方形の$\sigma = 0.72$ %（いずれも r=2.364 の場合)に比して小さいことがわかる．

３．４ 結言

本章では新しいディジタル測長法として”隣接２線走査法”（ＤＬＳ法）の原理を述べ，その測長特性について，対象図形の周囲長をとりあげ解析を行った．結果として画像フィールドを構成する最小単位である，画素の寸法を縦長に採ることによりその測定精度を大幅に向上可能なことを明らかにした．また実際にこの原理に基ずいたハードウエアを作成し，実際の円，矩形などの対象図形について周囲長，形状係数測定を行った．測定は矩形の姿勢変化，オフセット変化，円のオフセット変化，図形の面積変化などについて数多く行い，十分安定な測定結果が得られることを確認できた．またＤＬＳ法演算の実行部は非常にシンプルであり本手順のハードウエア化の容易さを実証している．３．３．３項（５）の実験結果は微小図形の高速判別への応用の可能性を示している．実験的検証のために今

回はハードウェア上で手順を実現したのであるが手順自体はハード，ソフトを問わない．ソフトウェアで処理する場合には相隣合う２本ではなく１つおきあるいは２つおきに取った走査線に注目して，これを１ラインずつずらして処理する方法へ発展させることもできる．これは４章にて詳述する．

第４章　飛越し走査を用いたディジタル測長

４．１　緒言

第３章で述べたように，ディジタル画像計測では指定された曲線に沿った２点間の長さ測定が必要になる場合がある．具体的には，濃淡画像の濃度等高線長や２値図形の輪郭線長の測定など，濃度階調の異なる領域の境界長測定がこれに相当する．ディジタル画像[2]の境界線は一般には画素間を線分で連結したディジタル曲線[2]で構成されるが，その線長測定では有限な画素の寸法，形状と画素間の位置関係を考慮しなければならない．本章ではこのような測定を高速且つ高精度で行う手法について述べる．第３章では画素を縦長の長方形に採って，測長精度を向上させる方法を示したが，これには画像入力ハードウエアに制約条件を付加することになる．本章では，この制約を必要としない，最も一般的に利用されている正方形画素のディジタル画像について議論を進める．

ディジタル測長法として従来から知られているものとしては第１章あるいは第３章で取り上げて説明した．３章で提案した隣接２線走査法も含めて，これらの方法は精度，処理速度，ハードウエア構成等の点でそれぞれ一長一短があり決定的なものではない．

本章では高精度で高速なソフトウエア処理が可能なディジタル測長法として，スキップ２線走査法（Skipped Dual Lines Scanning method 以下 SDLS 法と呼ぶ．）を提案する．この手法はソフトウエア処理に限定されたものではなくハードウエア化も可能である．本章ではまず測定原理を挙げ，輪郭が直線の場合の適用例を示す．次に直線について SDLS 法による測長評価式を導き，理論的な誤差を求める．また円弧について２値化からSDLS法による測長値を得るまでの手順を明確にする．実際の図形に対する測長特性を把握するため，輪郭が直線及び円弧の場合についてシミュレーションを行う．これらの結果に基づき，従来手法との比較検討を行う．また SDLS 法の測長評価式の数学的な証明を行う．

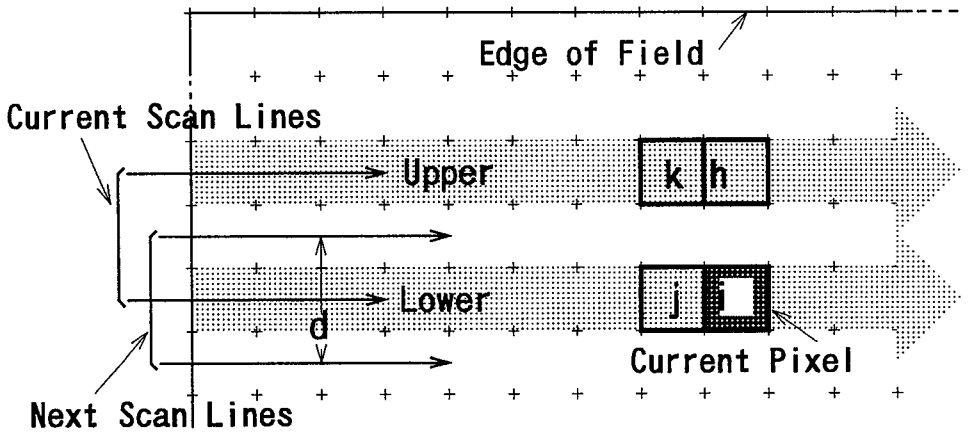


Fig.4.1　Scanning and pixels

４．２　ＳＤＬＳ　法による輪郭線測長原理

４．２．１　画像データのアクセス

画像データのアクセス方法は，ラスタースキャニング方式[2]である．２値化画像データを間隔 d （単位はPixel）の２本の走査線に注目してスキャンしながら，下記の手順により部分長を積算し，１フィールドスキャンの終了と同時に計測を終らせる．図４．１に手順の説明のための画素の対応関係を示す．i を今注目している画素，h は上の画素，j, k はそれぞれの一つ前の画素である． $s = d\text{-}1$ をスキップと呼ぶ．図４．２にスキャンによる画像データアクセス順序を示す．右上端から始めて①及び③の２本の走査線上をチェックして行く．（図４．２(a),(b),(c)）．この例は s=1 である．右端に到達したら１走査線だけ下にずらして走査を続ける（図４．２(d)）．４．２．２項に手順の詳細を示す．図４．３は手順をフローチャート形式で示したものである．

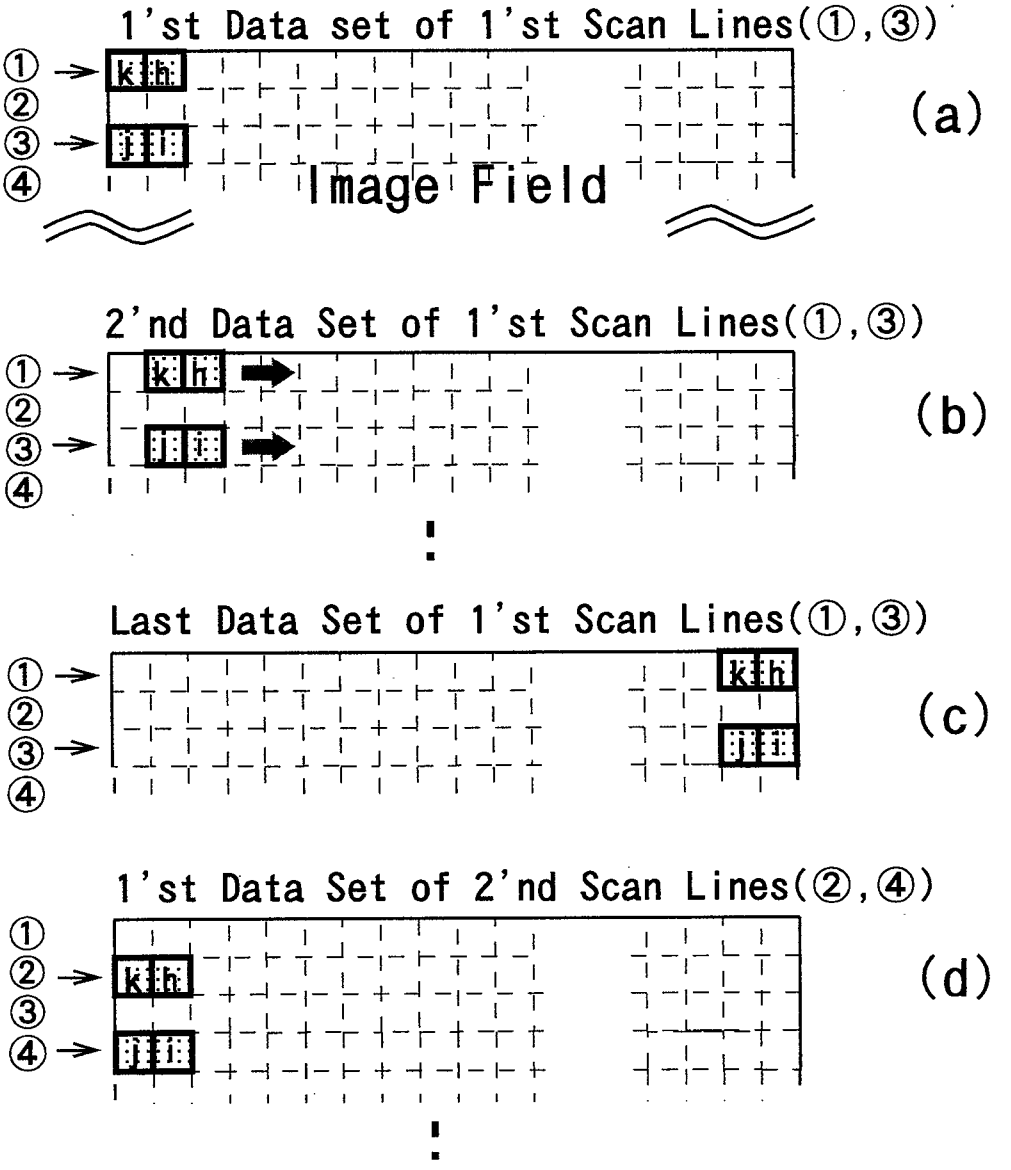


Fig.4.2 Image data scanning sequence

４．２．２ 飛び越し走査による測長原理

「まず m=0 とする．２値化後の２次元配列内メモリイメージに対して，間隔 d をおいて隣り合う２つの走査線上を，左から右へスキャンし，上下対応する画素値の排他的論理和が，連続して"１"である画素の数を数え（図４．３　"Ａ"グループ），この値を m とする．上下画素の排他的論理和が"０"で且つ，上下の画素のそれぞれ一つ前の画素との排他的論理和が，上下でいずれかが"１"である点（これを計算点と呼ぶ．図４．３　"Ｂ"グループ）に到達したら，$\sqrt{d^2+m^2}$ を計算し，あらかじめ用意した部分長累算部（図４．３　Acc）にたし込む．この後 m を０にクリヤし，さらに右方へ同じ操作を続ける．画像フィールドの下端に到達したら累算部の内容を d で除して測長値とする．」

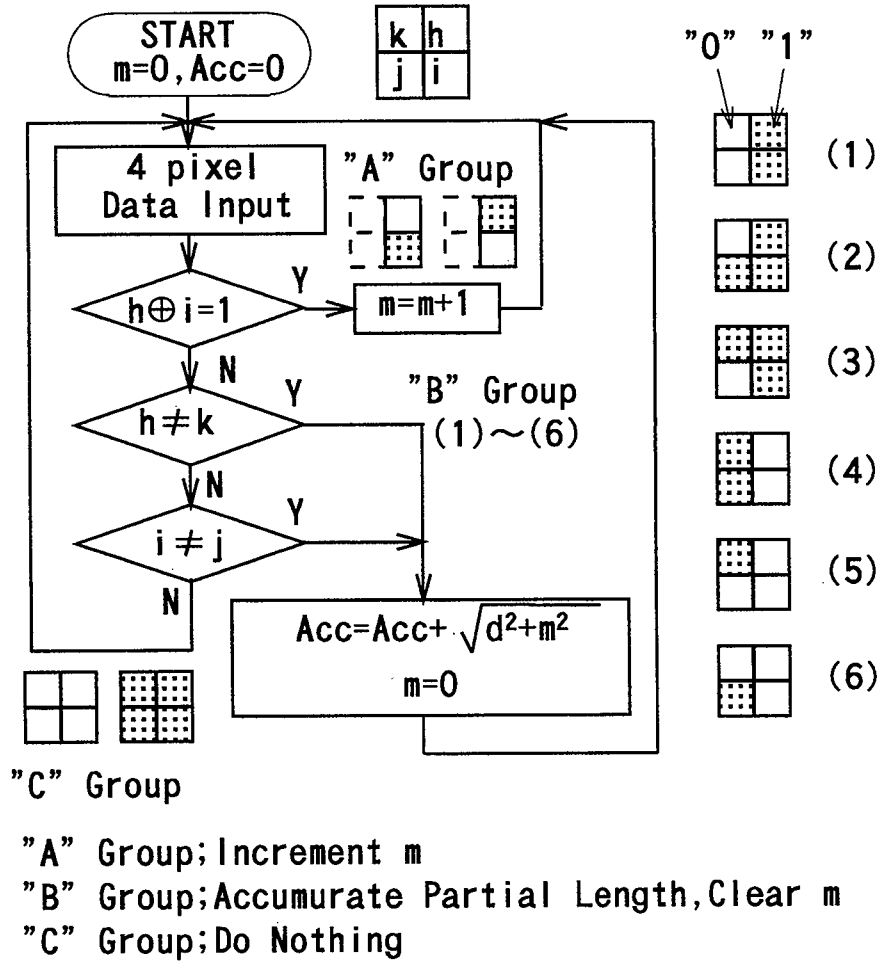


Fig.4.3　SDLS algorithm

計算点は論理式で書けば($\overline{h \oplus i}$)•(k⊕h + j⊕i)= "1" が成り立つ点である．図４．３の右端にも示しているが計算点のパターンは6個ある．これを輪郭推定線の対応付けを含めて図４．４に示す．図４．３では簡単のため s=0 としているが図４．４ではスキップを入れて s=1 として描いている．すなわち d=2 ，s=d-1=1 である．図４．４で(1)と(4)はそれぞれ図形の元図形の左側端，右側端の輪郭推定線に対応する．(2),(3)はそれぞ

れ元図形の上部及び下部の輪郭推定線に対応する．(5),(6)はそれぞれ元図形下部及び上部輪郭推定線に対応する．(1),(2),(3)と(4),(5),(6)のパターンはの互いに補の関係にある．この関係は３章に述べた隣接２線走査法と全く同じである．

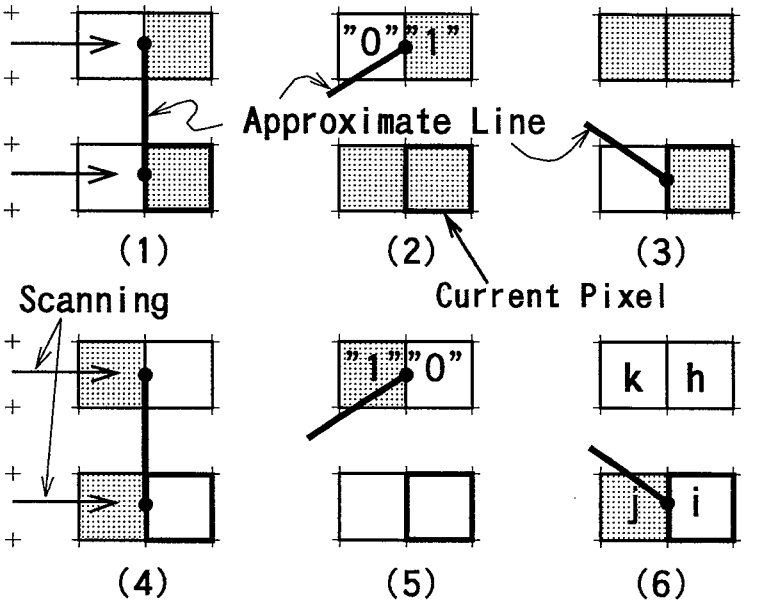


Fig.4.4　6 Pattern of calculating　timing point.

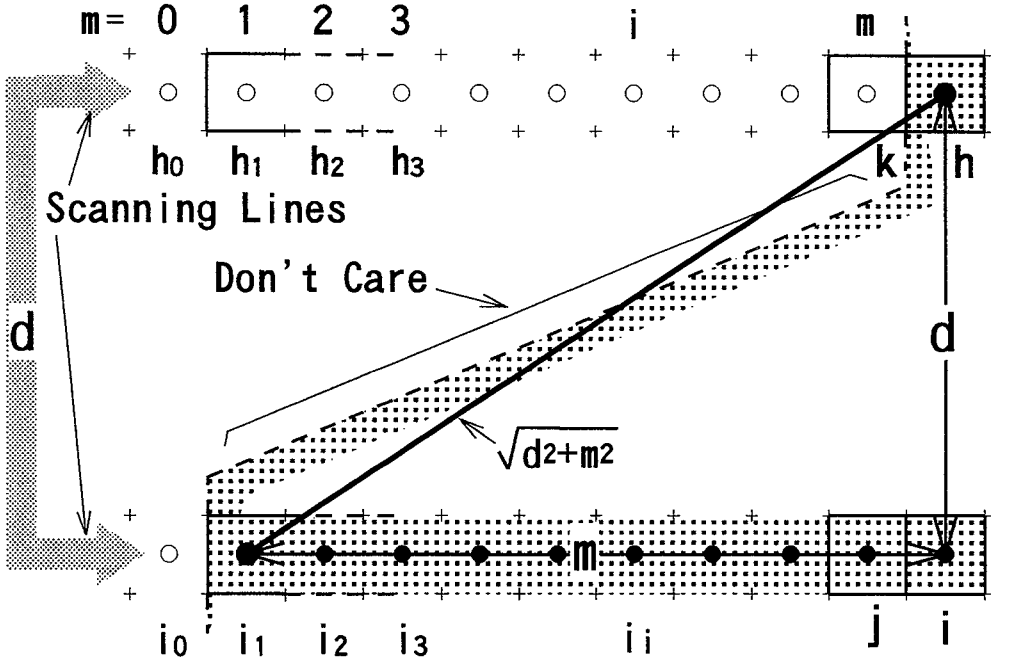


Fig.4.5　Calculation of partial length

図４．５に m のカウントと$\sqrt{d^2+m^2}$ の対応付けを示す．$h_i \oplus i_i$="1" が成り立つ点で

順次 m が＋１される（最上部に表示）．これは図形のエッジを検出する操作である．右端で計算点のパターン（２）が現れ $\sqrt{d^2+m^2}$ が計算される．これを図の太実線のように対応付けする．これを部分長と呼ぶ．部分長が接続されて図形の推定輪郭を構成する．

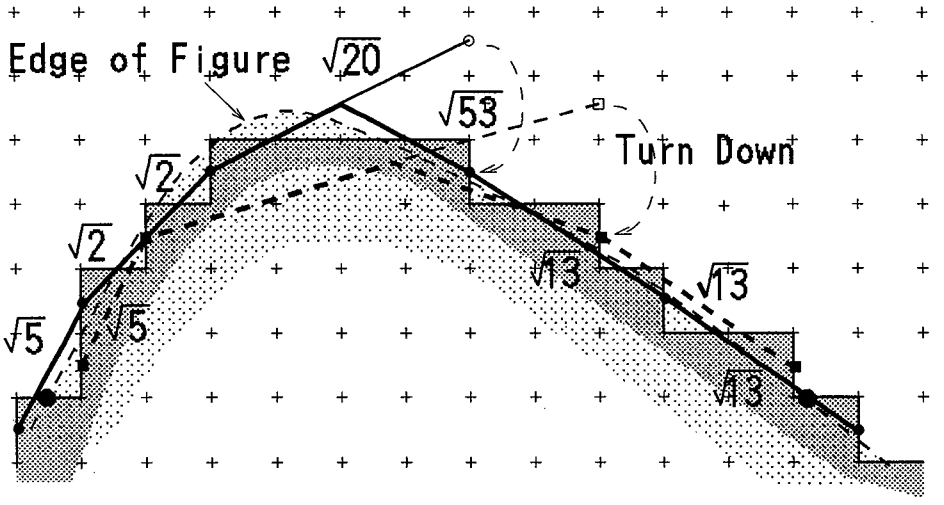


Fig.4.6 Example of edge length measurement.

図４．６は自由曲線に対する手順の実行例である．黒小丸間を結ぶ実線は，奇数番目のスキャンにより生成される部分長である．黒四角間を結ぶ破線は，偶数番目のそれである．これが累算部に累算される．図形が上下方向に凸の部分では，部分長を示す直線を中央で折り返して(Turn down)”輪郭”に対応付けする．自由曲線に沿った大きい黒丸間の長さは約 15.5 Pixelである． SDLS 法による測定値は 14.9 Pixel である．

上記の手順で，計算点における判断に必要なものは規則的に配列された，h, i, j, k ４個のデータである．そして m の計数と論理判断を繰り返す単純な構造であって，総データアクセス回数も従来の手法に比して少ない．これらは本アルゴリズムをハードウエア化する際の回路設計の容易さと，高速処理の条件に適合する．ハードウエアで実時間処理する場合，i は現データ，j は１画素の遅延データとして得られ，h, k は走査線遅延メモリを d 個用意することにより得られるので，１フィールド分のメモリを必要としない．また $\sqrt{d^2+m^2}$ の計算には表参照方式が利用できる．１方向からのスキャンで測定を行うため，画像入力と演算を並列に実行できるので，画像入力終了直後に結果が出る．このため移動物体の実時間形状計測に適している．さらに，この場合は垂直スキャンが不要なので，分解能の高い１次元イメージセンサを利用することができることも有利な点である．

４．２．３ 直線のディジタル測長

図４．７は傾き 3:5の無限長直線輪郭線を２値化してできたメモリイメージに，SDLS 法を適用して，輪郭線長を求めるときの様子を示す．ここで 3:5 はY方向：X方向＝ 3:5 を意味する．今後このように表す．傾き 3:5 の単位区間は２値化されて 1:2, 1:1, 1:2

3個，2種類の単位ステップに分解される．これをスキップ s=1（間隔 d=2）として

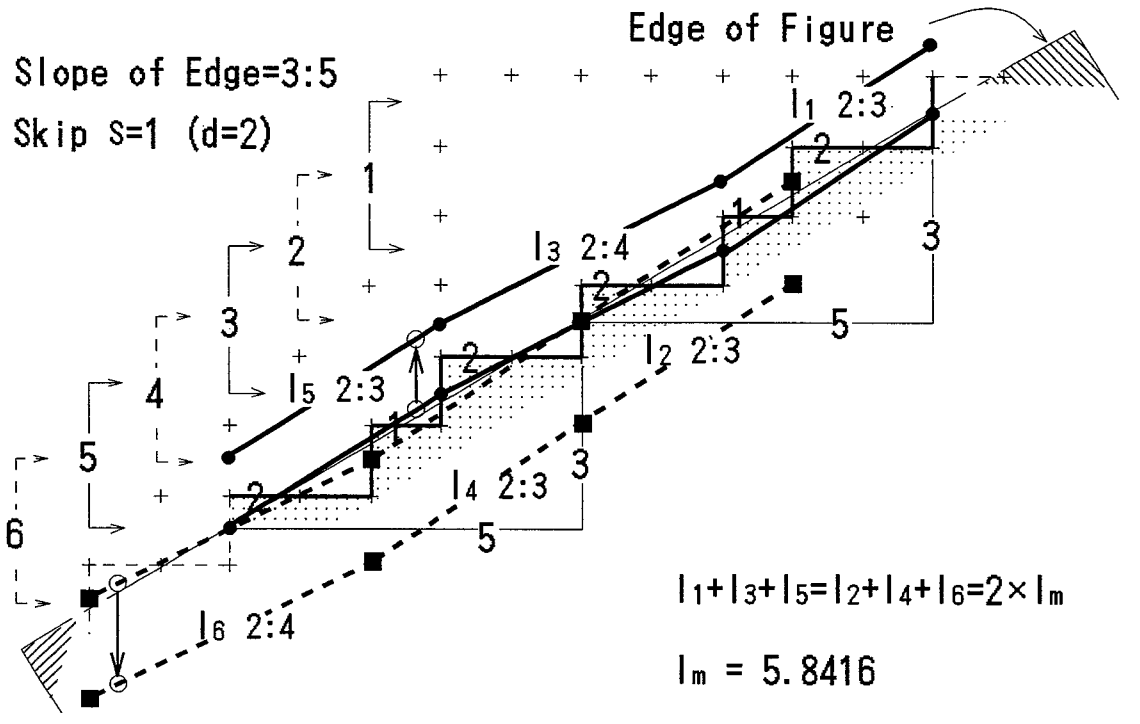


Fig.4.7 Example of edge length measurement for straight line edge

スキャンすると積算される部分長は l_1, l_2, l_3 ••• となる．スキャンする2本の走査線の位置を図左上方部に1，2，3・・・で示している．奇数番目のスキャンに対応する部分長を黒丸間を結ぶ実線，偶数番目のそれを黒四角間を結ぶ破線で示している．図が輻輳するのでそれぞれ上，下に取り出して描いている．この例では傾き 3:5 の区間2個をもって部分長生成の1周期を構成する．その後はこのパターンを周期的に繰り返す．そして第2番目の連鎖（破線） l_2, l_4, l_6 は1行だけずれて追随する．但し，同じ長さの部分長が現れる順序は異なる．部分長連鎖1周期の長さは $l_1+l_3+l_5=l_4+l_6+l_2$ である．傾き 3:5 の1区間に対応する長さを l_m とすると

$$l_m=(l_1+l_3+l_5)/2=(\sqrt{2^2+3^2}+\sqrt{2^2+4^2}+\sqrt{2^2+3^2})/2$$
$$=5.8416$$

となる．真の長さ l_t は

$$l_t=\sqrt{3^2+5^2}=5.831$$

となり相対誤差は 0.18 %である．図4．8は図4．7と同じメモリイメージにスキップ s を変化させてSDLS法を適用した様子を示す．図を明瞭にするため Y 方向に伸張して示している． s=0(d=1)のとき1個の部分長連鎖を生じる．連鎖中の繰り返しパターンの周期は元エッジの傾き 3:5 の1区間(A)である． s=1(d=2) では部分長連鎖は2個となり，パターンの周期は3:5の区間2個分(6:10)となる．一般的にはこのような部分長連鎖の1

周期はスキャンの間隔を d とするとき，傾き b:a（b, a は正の整数）の１区間の d 倍となる．またこの区間において d 個の部分長連鎖を生じるがその長さは同じである．これは厳密に証明可能である．図の右下，スキップ s=2（d=3）では黒丸を結ぶ連鎖，黒四角を結ぶ連鎖，黒三角を結ぶ連鎖と，連鎖が３個あるが平行している．傾き 3:5 の 3 と d の値が一致するとこのようなことが起こる．

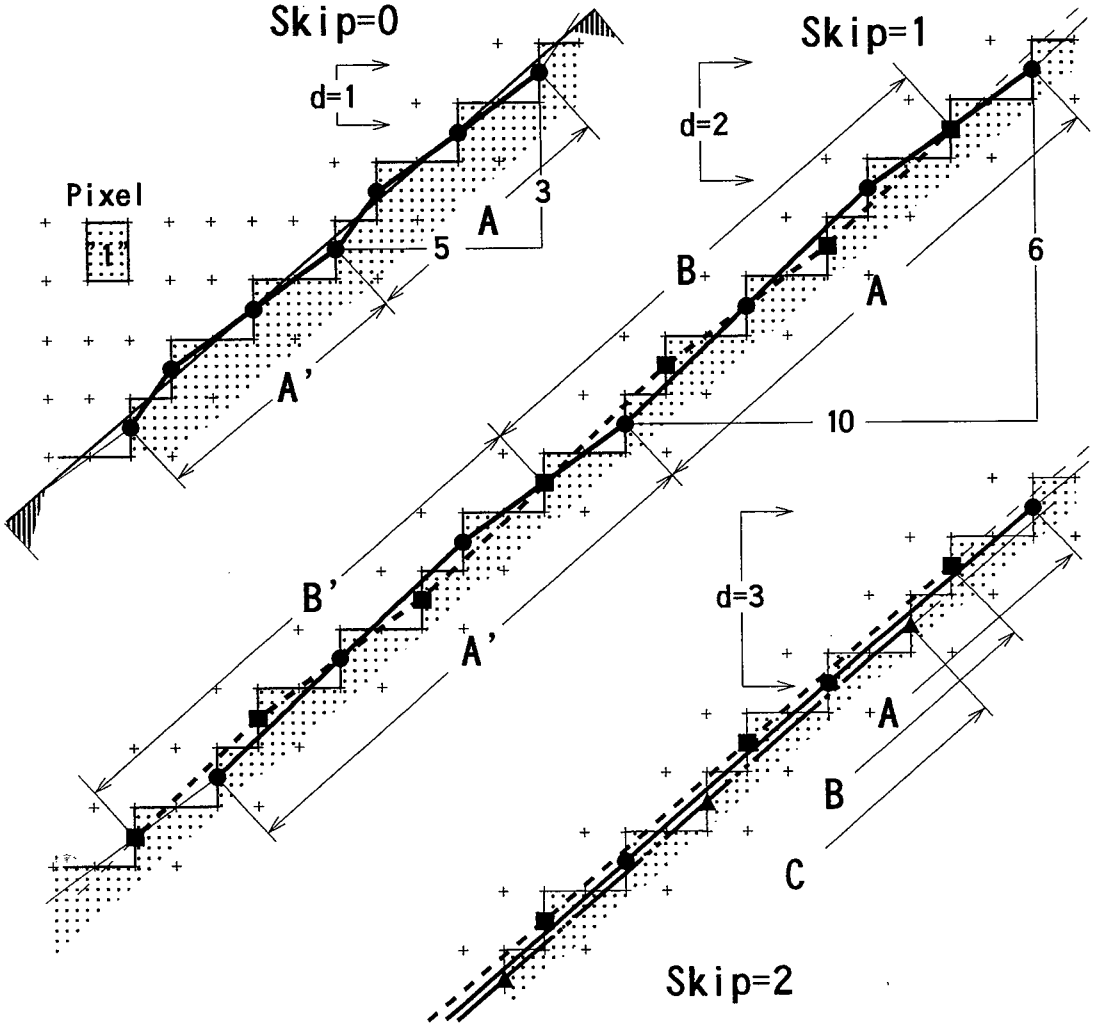


Fig.4.8　Example of edge length measurement (For different skip)

図４．９は直線エッジの傾きを変えて SDLS 法により測長した別の例である．図中(ａ)，(ｂ)共にスキップ s=1（d=2)としている．（ａ)では直線の傾きを 4:7 にとっている．測定される長さ l_m は黒丸間を結ぶ細実線 l_1, l_3 と黒四角間を結ぶ細破線 l_2, l_4 の長さを平均したものとなる．つまり

$$l_m=\{(l_1+l_3)+(l_2+l_4)\}/d=(\sqrt{4^2+2^2}+\sqrt{3^2+2^2}+\sqrt{3^2+2^2}+\sqrt{4^2+2^2})/2$$
$$=8.077$$

これに対して真の長さ l_t は

$$l_t=\sqrt{4^2+7^2}=8.06$$

である．相対誤差は　0.2%である．図４．９(ｂ)は直線の傾きを45°以上にとって測長した例で，もととなる直線エッジの傾きは 8:3 である．測定される長さ l_m は図(ａ)と同様に黒丸間を結ぶ直線 l_1, l_3, l_5, l_7 と黒四角間を結ぶ直線 l_2, l_4, l_6, l_8 の平均である．

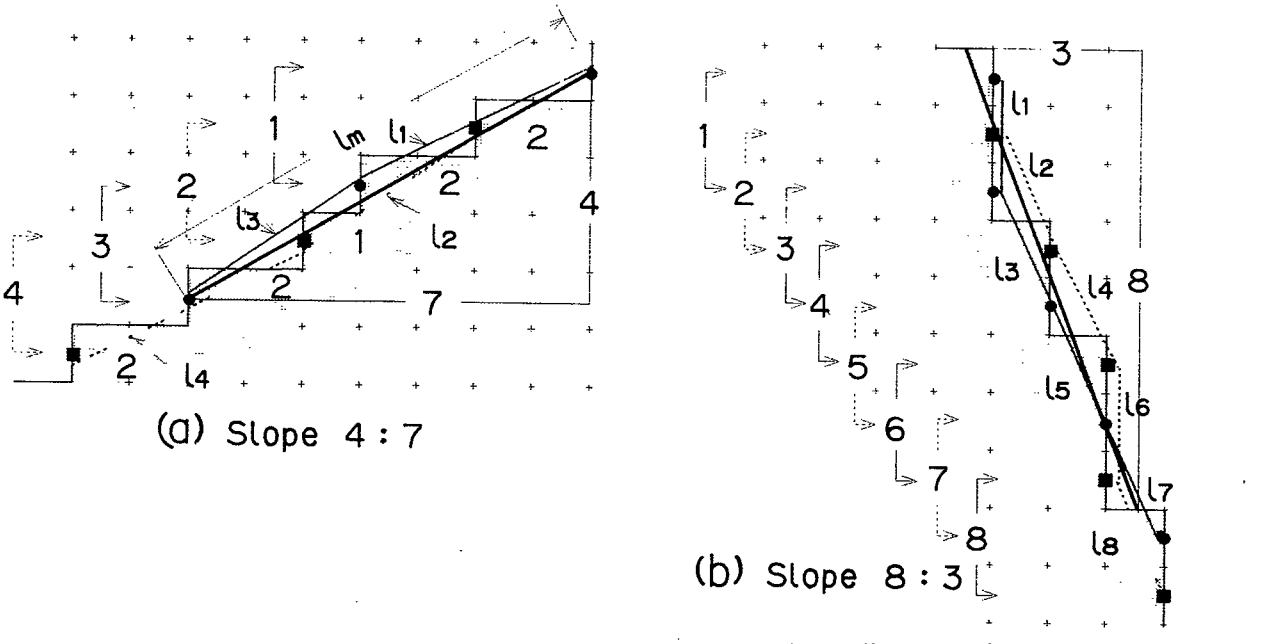


Fig.4.9　Edge length measurement by SDLS method for different slope

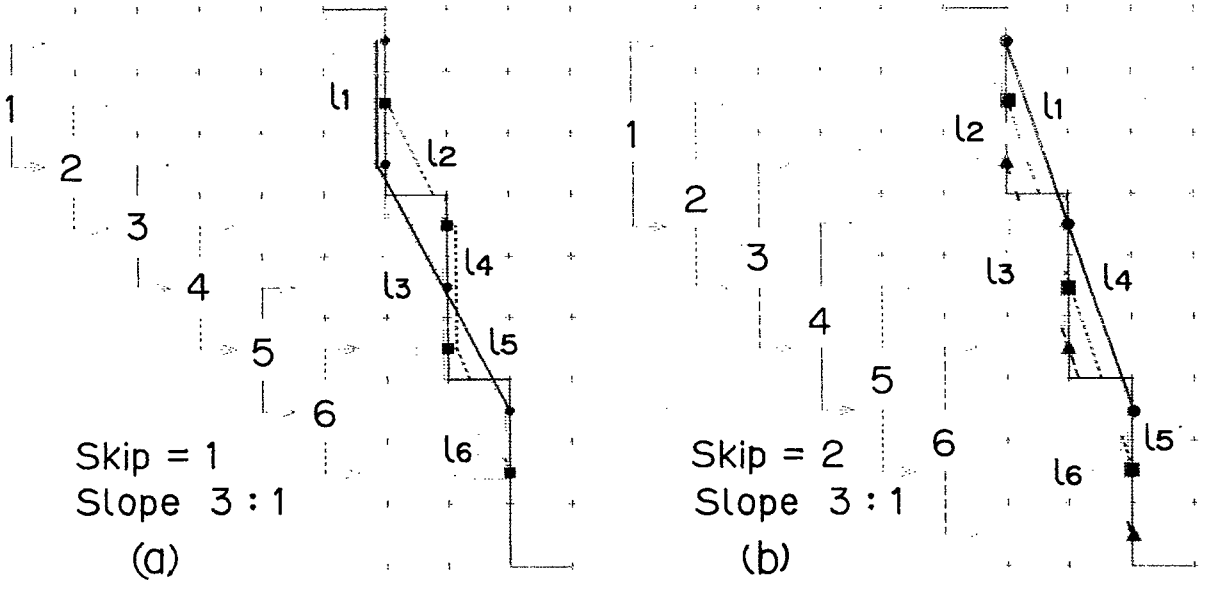


Fig.4.10　Edge length measurement by SDLS method for slope ＞ 45 ° (Skip s is varied)

つまり

$$l_m=\{(l_1+l_3+l_5+l_7)+(l_2+l_4+l_6+l_8)\}/2$$

$$=2+\sqrt{1^2+2^2}\times 3=8.7082$$

これに対して真の長さl_tは

$l_t=\sqrt{3^2+8^2}=8.544$

であり相対誤差は1.9%である．図４．１０は傾きが 45° を越える直線エッジに対してスキップ s を変化させてSDLS法を適用し測長した例である．傾きは図(ａ)，(ｂ)共に 3:1 にとっている．図(ａ)を見ると測長線は明らかにジグザグの度合いが激しく誤差の増大が予想される．実際計算してみると測長値 l_m は

$l_m=(2+\sqrt{1^2+2^2}\times 2)/2=3.2361$

であり，真の長さl_tは

$l_t=\sqrt{1^2+3^2}=3.16$

となる．相対誤差は2.4%であり，これまで挙げた例の中で最も大きい．一方同図(ｂ)ではスキップ $s=2$で走査間隔 $d=3$ の 3 と傾き 3:1 の 3 が一致して，図４．８に挙げたものと全く同じ現象が発生しており，測長線は元図形のエッジと完全に一致し，測長誤差は 0 となっている．

４．３　直線輪郭の２値化と部分長の生成

２章に述べた如く，長く続く傾き $b:a$ の直線輪郭が２値化されると２種類の単位ステップで構成される階段状のステップを生じる．これを SDLS 法により，間隔 d でもってスキャンしたときに生じる部分長 $\sqrt{d^2+m^2}$ について考察する．ここで部分長を表す線分の傾き $d:m$ (d,m は正の整数)を部分傾きと呼ぶことにする．２値化の際の画素の値は図形が当該画素の 50 %以上を覆っている時に”１”と判定する．図形の輪郭が直線の場合は，直線が画素中心の上下いずれを通るかで ”1”/”0” が決定される．この詳細は２章で述べている．

部分傾きの種類と個数に関して次の定理が成り立つ．

定理(1)

傾き $b:a$ の直線輪郭が２値化されたときのイメージを，間隔 d でスキャンするときに現れる部分傾きは ”$d:m$ だけ”，か ”$d:m$ と $d:m+1$ の２種類” かのいずれかである．ここで m は ad/b の整数部，すなわち $m=[ad/b]$である($m\geqq 0$)．また $b:a$ の区間を連続して d 個取ったときに（$bd:ad$ の区間），現れる部分傾き $d:m$ と $d:m+1$ の個数をそれぞれ p,q 個とすると，

$$p=b(m+1)-ad \qquad \text{---(4.1)}$$

$$q=ad-mb \qquad \text{---(4.2)}$$

であり，この式は d 個の部分長連鎖全てについて成り立つ．但し部分長連鎖があらわれる順序は同じとは限らない．この定理の証明は４．７．４項で行う．

４．４　ＳＤＬＳ 法による直線の長さ評価

定理(1)により，傾き $b:a$ の直線輪郭を２値化したメモリイメージを，間隔 d でスキャンするときに現れる部分傾きの，種類と個数が明らかになった．これより SDLS 法によ

り測定される区間 $b:a$ に対応する長さ l_m は

$$l_m = \{p\sqrt{d^2+m^2}+q\sqrt{d^2+(m+1)^2}\}/d \qquad ---(4.3)$$

となる．図４．７についてこれを確認すれば

$b:a = 3:5$ より $b=3, a=5$ また $d=2$

定理(1)より $m=[ad/b]=3$

$p=2,\ q=1$

すなわち 3:5 の２区間に 2:3 を２個，2:4 を１個生じる． そして l_m は式(4.3)より

$l_m=\{2\sqrt{2^2+3^2}+\sqrt{2^2+4^2}\}/2=5.8416$

真の長さ l_t は

$l_t=\sqrt{3^2+5^2}=5.8310$

となり折れ曲がって近似されるため，測定値は正の誤差を示すことがわかる．

図４．１１は横軸に，もとの輪郭線の傾き $\theta=tan^{-1}(b/a)$ をとり，スキップ s をパラメータとして，縦軸に誤差 $(l_m-l_t)/l_t\times100$ % を表している．$45^\circ<\theta<90^\circ$ の領域で誤差が大きいが，スキップを行うことにより減少することがわかる．図中に角度についての平均誤差（Average Error）を示す．

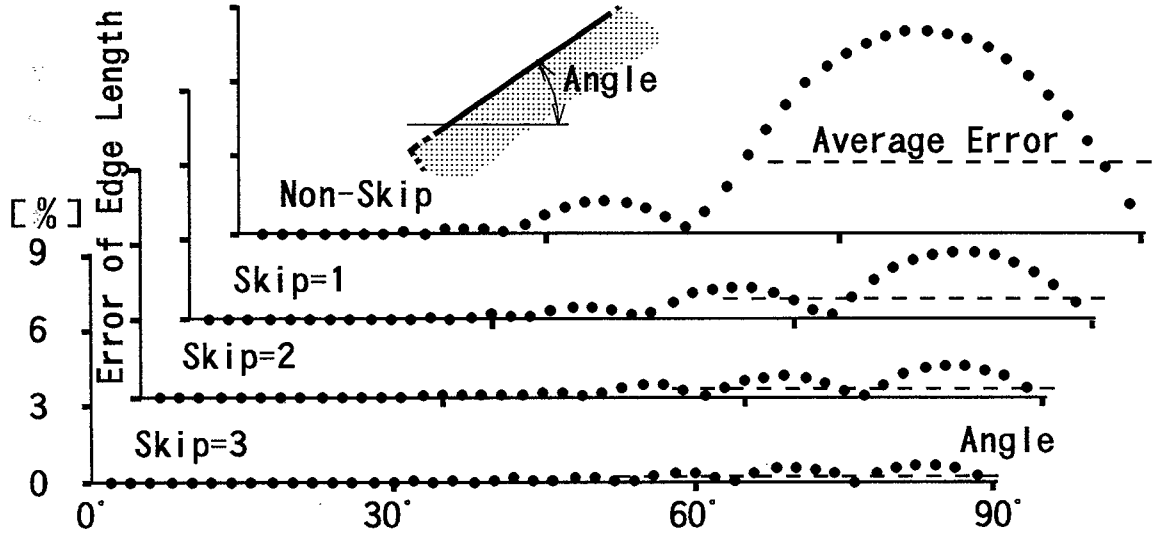


Fig.4.11 Error of measured edge length.

４．５ 円弧輪郭の２値化と SDLS 法による測長値

直線に比較して円弧を規定するパラメータは多い．半径 r，中心座標 x_c，y_c，始角 θ_1，終角 θ_2 の５つがある．円弧の２値化については直線の場合のように，ある程度の長さをとれば同じパターンの繰り返し的な処理が不可能である．そこで便宜的に円周を考え平均的な弧長として取り扱うことにする．さて r，x_c，y_c が決定した円に対し，SDLS 法により，ある手順のもとで周囲長が一意的に決定できる．

この関係は(4.3)式のように解析的な式で表現することは困難であるが手順として確立している．これを次に述べる．

仮想円の周長 l_mを求める過程は直線の場合と同様に(1)図形の２値化，(2)SDLS法による測長，に分けられる．Poを原図のパターンとし，これを入力として２値化イメージ Pbを出力する関数をFbとする．すなわち

$$Pb = Fb(Po)$$

この Pbに SDLS 法を適用して周長 l_mを求める操作をFsとする．すなわち

$$l_m = Fs(Pb)$$

Fsはすでに述べたように確立している．Fbは円が各画素を覆う面積を幾何学的に計算する作業である．これは２章にて詳述した．かくして r, x_c, y_cで示される円の周長は

$$l_m = Fs\{Fb(Po)\}$$

として求まる．Fsはスキップ s を，Poは r, x_c, y_cをパラメータに持つ． x_c または y_c が丁度１画素の整数倍ずれると l_m は同じ値をとる． r が 10～11 画素以内で変化したときの l_mの値の真値からの誤差（%）の計算例を４．６．３項の図４．１４に示すが規則的ではない．但しスキップを行うことによって誤差が減少している．

４．６　コンピュータシミュレーション

４．６．１　シミュレーションの対象及び方法

SDLS 法による輪郭線測長特性のコンピュータシミュレーションを行った．対象とする輪郭は直線と円弧である．端点付近の長さ定義の不安定性を排除するため閉曲線を対象とし，正方形と円の周囲を測定した．図形の２値化メモリイメージの作成は正方形については２．３節に述べた方法によった．円についても２．３節に述べた如く，各画素について円が覆っている面積を求めて"1"/"0"を決定した．

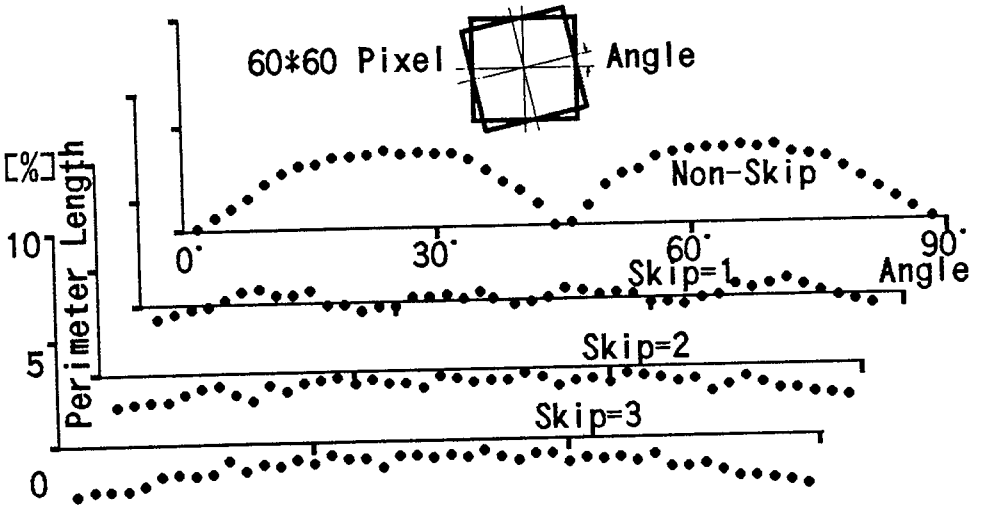


Fig.4.12　Error of perimeter length of square.

4．6．2　正方形についてのシミュレーション結果

正方形の場合，姿勢の変化による測定値のばらつきが多い（３章参照）．そこで次の順序でシミュレーションを行った．

(1)辺の長さ 60 Pixelの正方形の中心を座標原点に置き，(2)２値化データを作成し，(3)SDLS 法により周長を測定した．(4)正方形の中心を座標原点からX方向 0～0.4，Y方向 0～X の範囲で移動させ(0.1きざみ)，(2),(3)を繰り返し，平均値を求めた．(5)図形の姿勢を２度きざみで変化させて(1)～(4)を繰り返した．

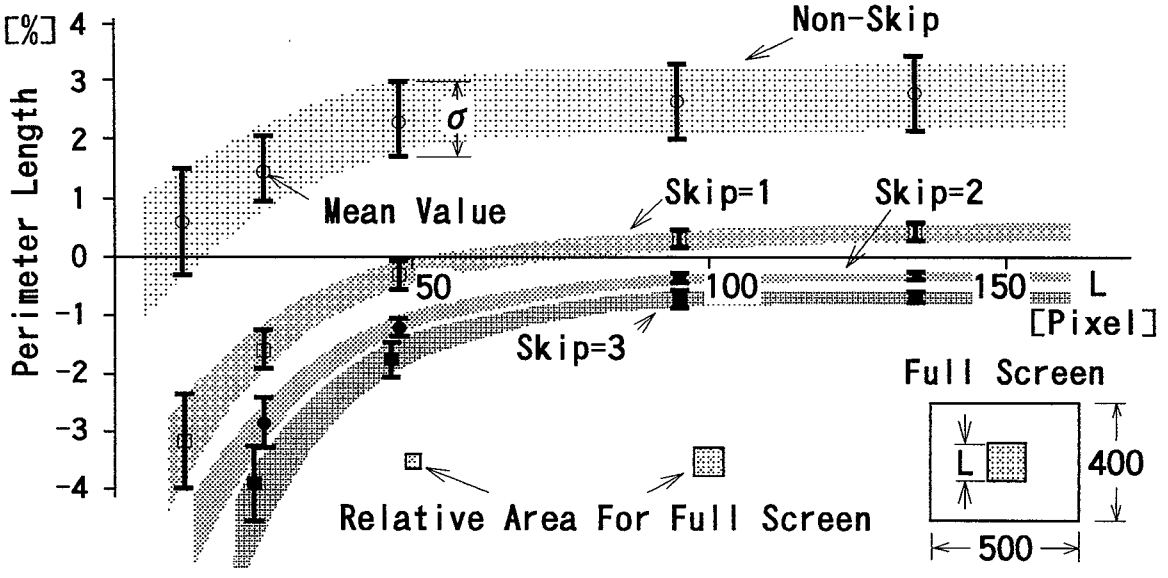


Fig.4.13　Error of perimeter length of square with different size.

この結果を図４．１２に示す．横軸は角度，縦軸は真の値からの誤差である．スキップを行うことにより姿勢の変化に伴うばらつきが減少し，平均誤差が小さくなる．しかし過度のスキップは負の誤差を生じる．

単純に測定系の分解能を上げれば測定精度は向上する．したがって測定手法の評価の際には分解能と対象図形の大きさの関連で精度を議論せねばならない．そこで次に対象図形の面積変化に対する測長特性を検討するため，以下の手順でシミュレーションを行った．

(1)辺の長さLの正方形の中心を座標原点から(0.3,0.1)Pixelだけずらした位置に置き，(2)２値化データを作成し，(3)SDLS 法により周長を測定した．(4)図形の姿勢を0°～90°の範囲で17種類変えて(2),(3)を繰り返し，平均値と標準偏差σを求めた．(5) Lを変化させて(1)～(4)を繰り返した．

この結果を図４．１３に示す．横軸はL，縦軸は真の値からの誤差である．L=25 Pixel 以上でスキップを行うことにより誤差が少なくなる．しかしスキップが多すぎると負の誤差が増す．図形が小さいときは負の誤差が多い．測定のばらつきはスキップを行った方が少ない．図の下方に視野全体の大きさ(500×400 Pixel)に対する対象図形の相対的な面積比を示す．

４．６．３　円弧についてのシミュレーション結果

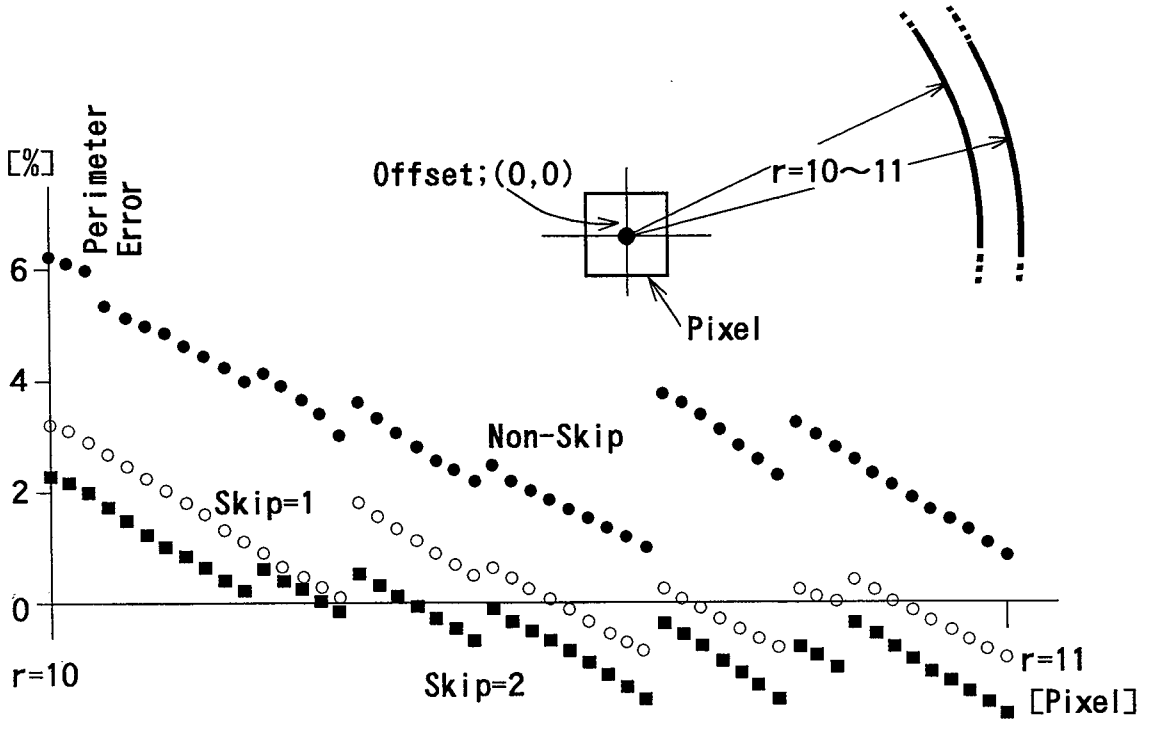


Fig.4.14　Error of perimeter length of circle (For different *r* and skip)

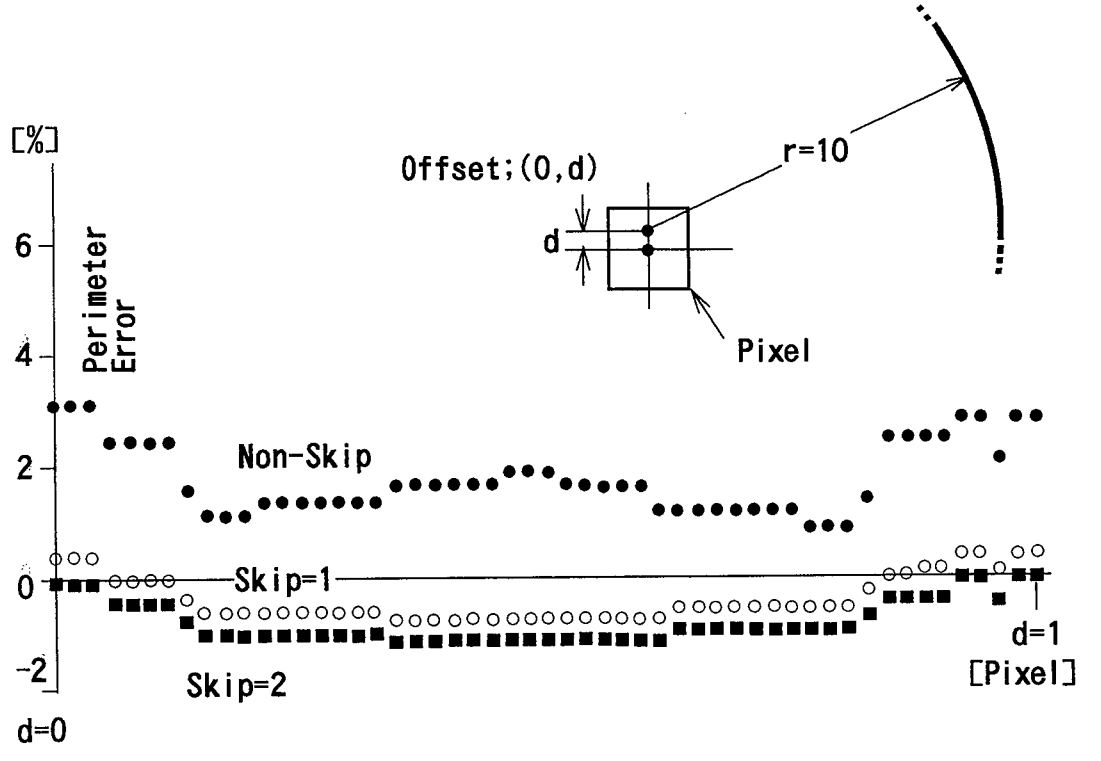


Fig.4.15　Error of perimeter length of circle with center offset

円弧について次のようなシミュレーションを行った．まず比較的，面積の小さな（約300 Pixel^2)について，その半径を一画素の範囲で変化させ SDLS 法による測長誤差を測定した．円の中心は画素中心に一致させた．図４．１４はこの様子を示している．横軸は半径 r ,縦軸は理論値からの誤差[％]である．半径は原点を r=10 として r=11 まで変化させた．オフセットが０なのである程度，周期をもって,誤差が変化している．スキップ s=0 を黒丸，s=1 を白丸，s=2 を黒四角であらわしているが， s=1 のときが最も誤差が少ない．

次に半径 r を一定としてY軸方向に画素中心と，円の中心の間にオフセットを持たせて測長した結果が図４．１５である．対象円の半径 r は１０[Pixel]である．横軸に画素中心とのオフセット d をとり，縦軸は理論値からの測長誤差[％]をあらわしている．測長値が一定の部分が継続している部分が認められるが，これは円を２値化したパターンが一定な状況を示している．スキップ量と精度の関係は前図と同様に s=1 で最も良好な結果が出ている．

次に図４．１４と同様に半径 r を r=10～11[Pixel] の範囲で変化させ，画素中心と円の中心の間にX方向0.1，Y方向0.2[Pixel]のオフセットを与えて測長した結果が図４．１６である．この場合はオフセットが非対称なため図４．１４に比して変化の周期が短くなっている．横軸は半径 r，縦軸は理論値からの測長誤差（％）である．スキップ s=1 で誤差が最も少ない傾向は変わっていない．以上３個の図では半径変化，オフセット変化，オフセットを与えての半径変化についてシミュレーションを行ったが,いずれに場合も s=1 のときが最も誤差が少ない．

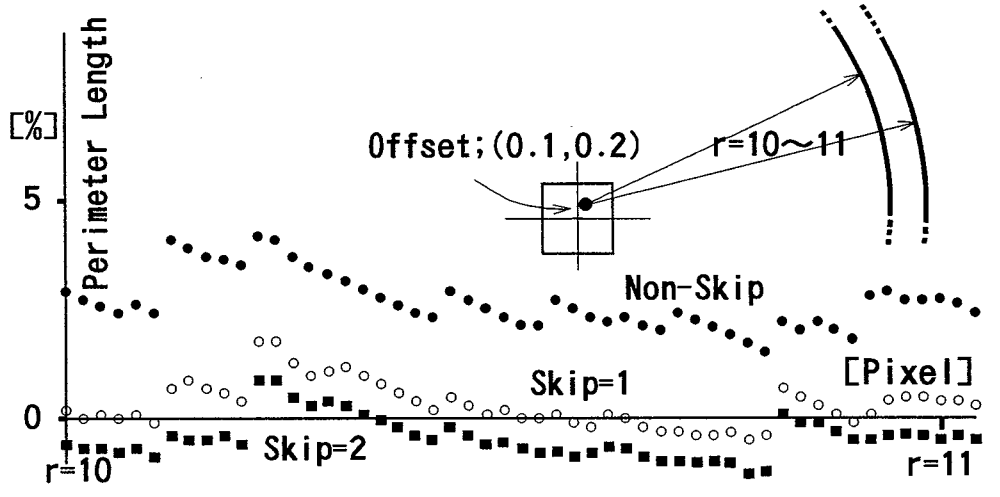


Fig.4.16 Error of perimeter length of circle with offset (x,y direction)

しかしこれは半径 10 Pixel（面積=314 [Pixel^2]）の比較的小さな面積の図形についての結果である．そこで次に面積が変化したときの，SDLS 法の測長特性を調査するため次のような方法で測長シミュレーションを行った．

(1)直径Dの円を座標原点に置き，(2)２値化データを作成し，(3)SDLS 法により周長を測

定した．(4)円の中心をX方向 0-0.5，Y方向 0-X [Pixel]の範囲でずらし，(2),(3)を繰り返し，平均値と標準偏差σを求めた．
(5)直径Dを変化させて(1)～(4)を繰り返した．

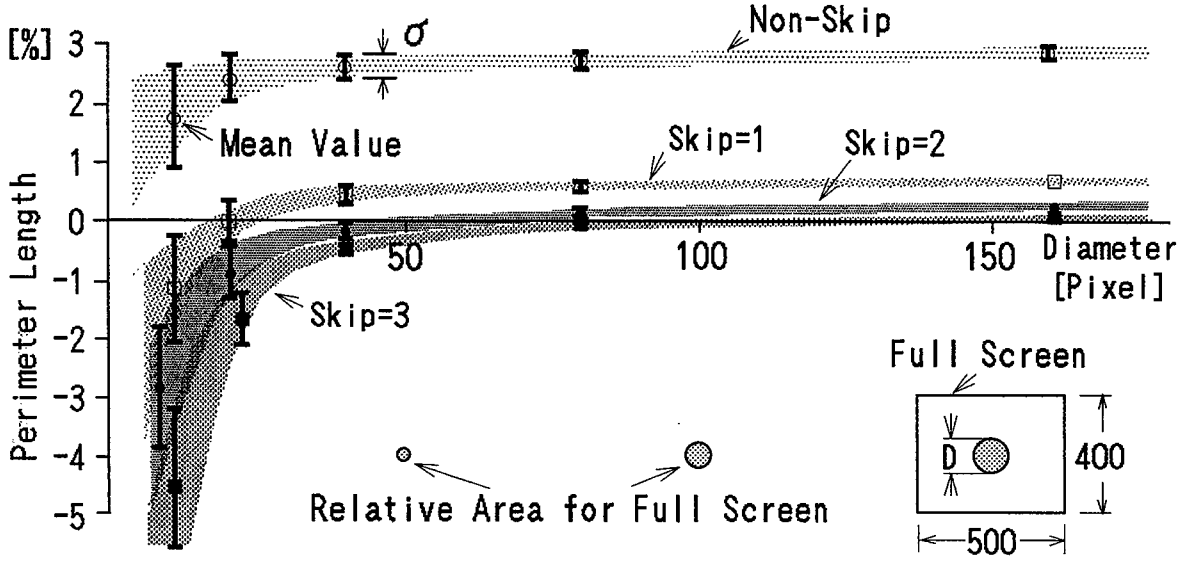


Fig.4.17　Error of perimeter length of circle with different size.

この結果を図４．１７に示す．横軸はD，縦軸は理論値からの誤差である．D=10 Pixel 以上でスキップを行うことにより誤差が少なくなる．全般的に正方形の場合と同じであるが，円の場合はスキップをさらに大きくしても誤差が負になる傾向は小さい．測定のばらつきはスキップを行った方が少ない．

４．６．４　微小図形の２値化とSDLS法による測長結果

図形の面積が 50 [Pixel^2] 程度以下になると元図形がどのようなものであったかの判別が困難になってくる．ここでは正方形と円について，図形が小さいときの両者の判別についてシミュレーションを行った結果について述べる．

図４．１８は一辺の長さ 7 [Pixel]，面積 49 [Pixel^2]の正方形をオフセットと回転を与えながら２値化したものである．左上隅の図が対象となる正方形の中心と画素中心を一致させたときのもの，これより下方向に，画素中心と正方形の中心を 0.3 [Pixel] ずつY方向オフセットを与えている．また右方向に進むに従って 10° ずつの回転を与えている．

また図４．１９は半径 r=4[Pixel],面積 S=50 [Pixel^2]の円をオフセットを与えながら２値化したものである．左上隅の図が円の中心と画素中心を一致させたときのものである．これより下方向にY方向オフセットを 0.3 [Pixel] ずつ加えている．また右方向に進むに従ってX方向に 0.2 [Pixel] きざみでオフセットを与えながら２値化を行っている．

これらそれぞれ 15 個の図形についてスキップ s=1 を与えた SDLS 法により，形状係数 F の測定を行った結果の平均値と標準偏差を表４．１に示す．正方形で F=1.15 ,円のと

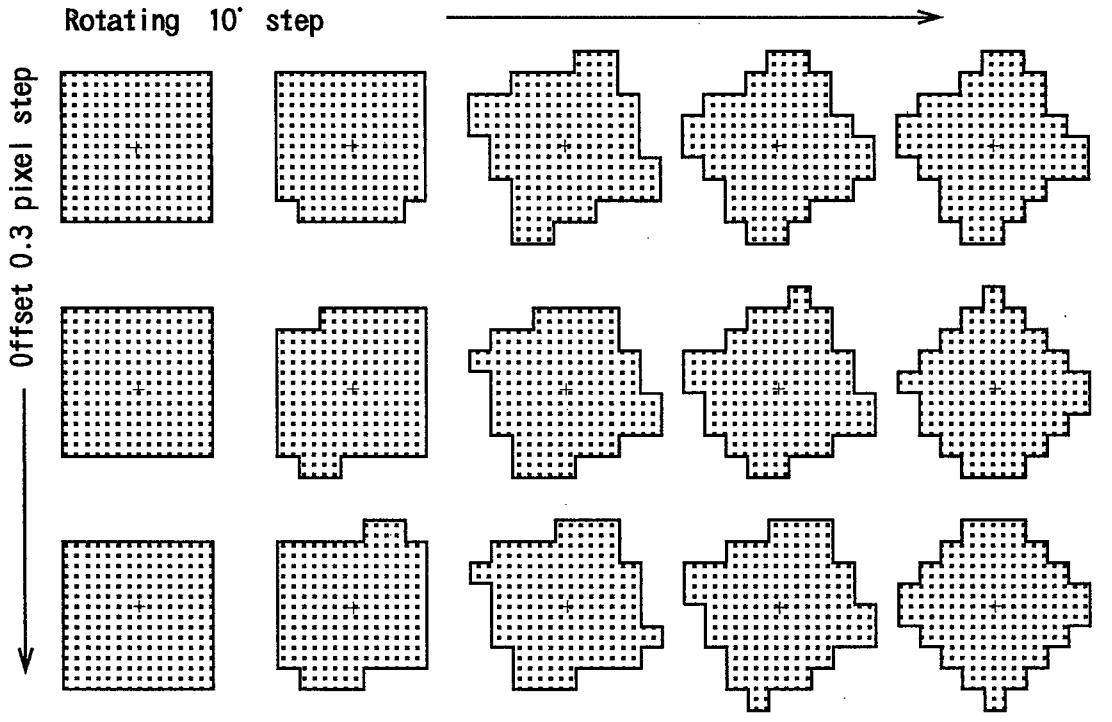


Fig.4.18 Binary image for small rectangular figure (Offset and rotation)

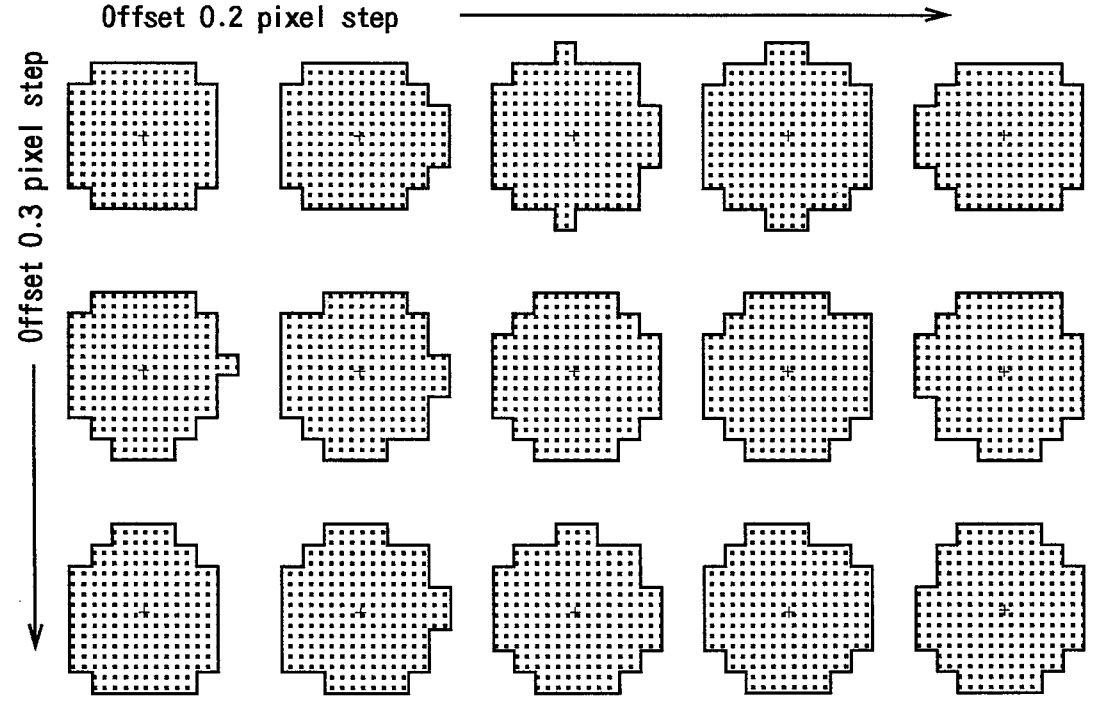


Fig.4.19 Binary image for small circular figure with offset x,y direction

き F=1.05 であり有為な差が現れている．この程度の大きさの図形では元図形が正方形であるか円であるかは人間でも目視確認が困難であるが，この例のように図形の姿勢あるいはオフセットを変えながら測定を繰り返し，その平均をとることによって判別が不可能でないことを示している．

Table 4.1 Shape factor for small rectangular and circle

	形状係数 （理論値）	標準偏差
正方形	1.115 (1.27)	0.075
円	1.05 (1.00)	0.032

4．7 考察

4．7．1 走査線スキップが測定精度の向上に及ぼす効果

スキップを行うことによる輪郭線測長精度向上の理由は第2章で述べた．ここではアルゴリズムの理解を踏まえた上で原画像エッジの傾きを変えて再度考察する．図4．20は傾き 2:1 の直線輪郭線にスキップ s= 1 として SDLS 法を適用したときの様子を示す．左図で奇数回目のスキャンにより積算される単位ステップ当りの長さ（実線）は中図の実線と同じであり，これは細かいメッシュでハッチされた画像フィールド F_1 上で 隣接2線走査法を適用した場合の長さに等しい．

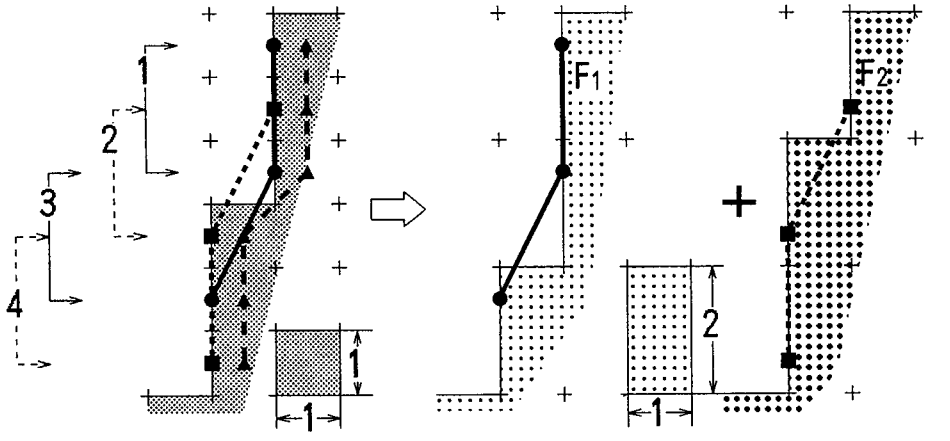


Fig.4.20 Original field is separated into two different field F_1 and F_2.

偶数回目のそれは破線と，太いハッチで示される F_2 である．F_1, F_2は画素の縦横比 r を 2:1 の比で縦長に採った場合と等価である．左図で三角の点を結ぶ破線がスキップを行わないときの部分長を示しているが画素が縦長の場合，輪郭近似直線の方向がもとの輪郭線の方向に沿うように変化し近似精度が向上する．

この点については3章に詳述している．3章では r の最適値として r = 2 前後を結論したがスキップ s =1 のときは r = 2.00 である．

4．7．2　測長特性とスキップ s の最適値

4．4節では SDLS 法で直線輪郭線を測長した場合，誤差は必ず正になることを示した．しかし正方形に対するシュミュレーション結果 図4．12 を見ると，スキップを増した場合，誤差が負の方向に漸次増大する．この原因はスキップにより縦分解能が落ち正方形の角が丸められるからである．このように図形の鋭利な変化がある部分では図形の空間周波数が，分解能によって決まる帯域幅に近い成分を含んでいるため誤差が増加する．図形が小さいときも同じである．

図形が円の場合には一般的に誤差及び，ばらつきが小さく，スキップ s を増やしても図形が大きい範囲では誤差が負にならない．このことは自然界に存在する物体の形状を測定する際，好ましい特性である．

円の場合も図形が小さい領域では先の理由で負の誤差が大きくなり s が大なるほど顕著であるから，最適な値が存在する．4．6節での議論と総合して s = 1 が最も適当な値である． s = 1 として，セラミック粉体の粒度解析に応用した場合，誤差を±1％とするなら視野内粒子数，約1000個まで適用可能である．（粒子を球と仮定し，図4．17より求めた．）

4．7．3　各種ディジタル測長法の比較

文献(3)～(5)を参照し本論文のデータと比較すれば SDLS 法は 1．2．2項で述べた(1),(2),(3),(4)の方法より誤差が少ない．また”3画素ベクトル法[4]”に比して微小図形を対象としたとき，精度は劣るが処理速度が速い．Hough変換から求める方法に比して誤差が少なく処理速度が速い．また使用メモリが少ない．3章で述べた”隣接2線走査法”と比較すれば SDLS 法は 図形が小さいときに若干誤差が増える傾向にあるが精度，実行速度共に同程度である．しかし隣接2線走査法には画素を長方形に採るという制約があるが，SDLS 法にはない．従来からある画像入力装置がそのまま使用可能という点が極めて有利である．また画像メモリの一部に限って適用できるのでポインティングデバイスと併用して，スクリーン上で図形の縁に沿った長さを容易に測定できる．

4．7．4　部分傾きの種類と個数に関する定理の証明

ここに定理(1)を再掲してその証明を行う．

定理(1)　傾き b:a の直線輪郭が2値化されたときのイメージを，間隔 d でスキャンするときに現れる部分傾きは ”d:m だけ”，か ”d:m と d:m+1 の2種類” かのいずれかである．ここで m は ad/b の整数部，すなわち m=[ad/b]である($m \geqq 0$)．また b:aの区間を連続して d 個取ったときに（bd:ad の区間），現れる部分傾き d:mと d:m+1の個数をそれぞれ p,q 個とすると，

$$p=b(m+1)-ad \qquad \text{---(4.4)}$$

$$q=ad-mb \qquad \text{---(4.5)}$$

であり，この式は d 個の部分長連鎖全てについて成り立つ．但し部分長連鎖があらわれ

る順序は同じとは限らない.

証明

部分傾きともとの輪郭の存在範囲に関して次の補題が成立する.

補題(1) 間隔 d でスキャンする際に,部分傾き $d:m$ が現れたとすると, もとの直線 輪郭線の傾きを $b:a$ として次の関係が成り立つ.

$$d/(m+1) < b/a < d/(m-1) \qquad ---(4.6)$$

または

$$m-1 < ad/b < m+1 \qquad ---(4.7)$$

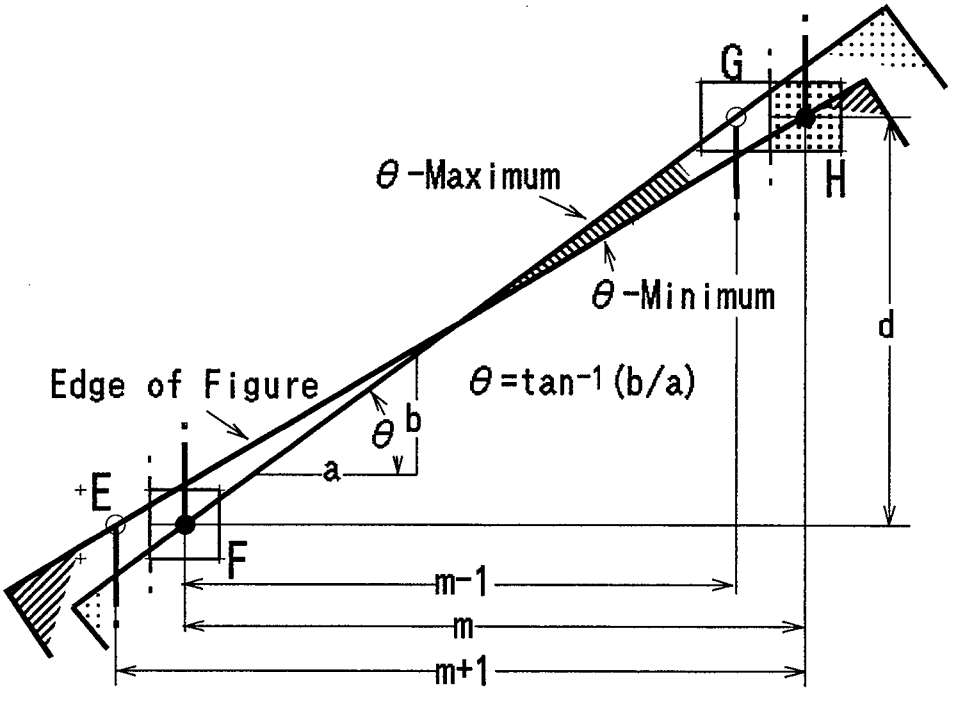


Fig.4.21 Binary image and original straight lines

補題(1)の証明

図４．２１のように $d:m$ の傾きが現れたとすると

画素Eは"0"であるから直線は E の中心より下を通る.

画素Fは"1"であるから直線は F の中心かそれより上を通る.

画素Gは"0"であるから直線は G の中心より下を通る.

画素Hは"1"であるから直線は H の中心かそれより上を通る.

したがって傾き最大の直線を考えると $b/a < d/(m-1)$

傾き最小の直線を考えると $b/a > d/(m+1)$

これより $d/(m+1) < b/a < d/(m-1)$

となり式(4.6)が得られる. 両辺の逆数をとり d をかければ(4.7)が得られる.

補題(1)を使って定理(1)が証明できる.

定理(1)の証明.

m が与えられたとき $d:m$ のステップを生じる，直線の傾きの逆数の d 倍 ad/b （ここでは除算の結果で考える．このことを明示するため：ではなく／を用いる．）の集合P_mは式(4.6)によって数直線上で 図４．２２ ② の如く示される．両端は空である．同様にして $d:m+2, d:m+1, d:m-1$に対応する $P_{m+2}, P_{m+1}, P_{m-1}$は④，③，①で示される．

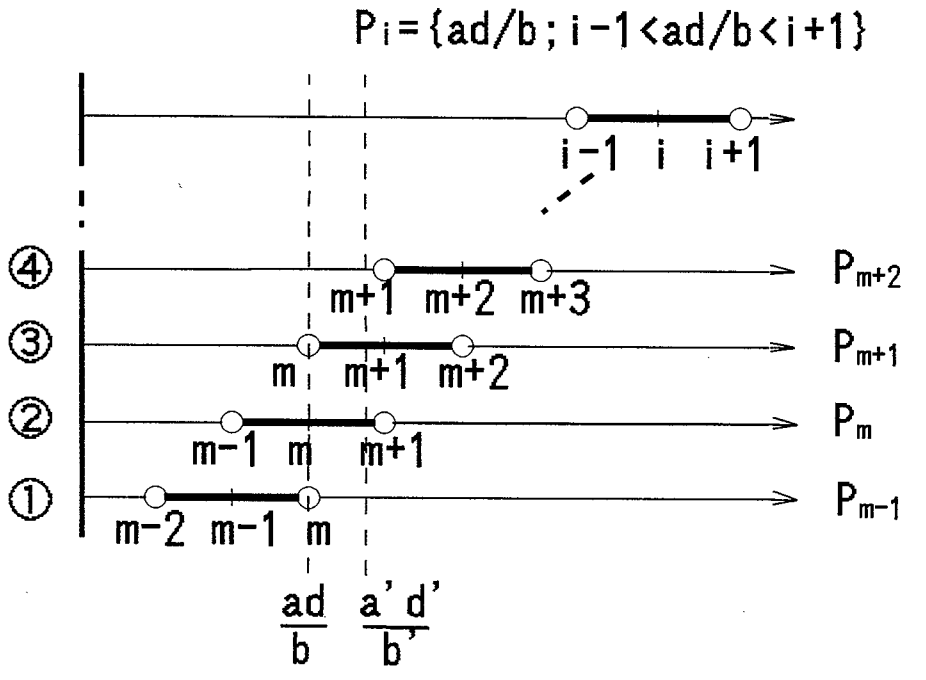


Fig.4.22 A set of reciprocal of ad/b on numerical line

P_iは一般項である．

図より数直線上の特定の一点はP_i の内の１つに属するか，または 連続するP_i，P_{i+1}に属することがわかる． したがって傾き $b:a$ が与えられたときに生じるステップは”$d:i$だけ” か，”$d:i$ と $d:i+1$” のいずれかである．

図中破線で示した ad/b はP_mだけに属し，数直線上の位置関係から m は ad/b の整数部である．また $a'd'/b'$はP_m とP_{m+1} に属するが m は $a'd'/b'$の整数部である．

次に d 個の部分長連鎖の一つについてX方向への移動量 adと，Y方向への移動量 bd を考えると(図４．２３参照)

$$ad=pm+q(m+1)$$

$$bd=pd+qd$$

これより p, q を求めると式(4.4),(4.5)が得られる．$q=0$ のときは”$d+i$だけ”に対応し $d:m$ のステップのみを生じる．以上の証明は特定の部分長連鎖に限られたものではない．従って d 個の部分長連鎖について成り立つ．

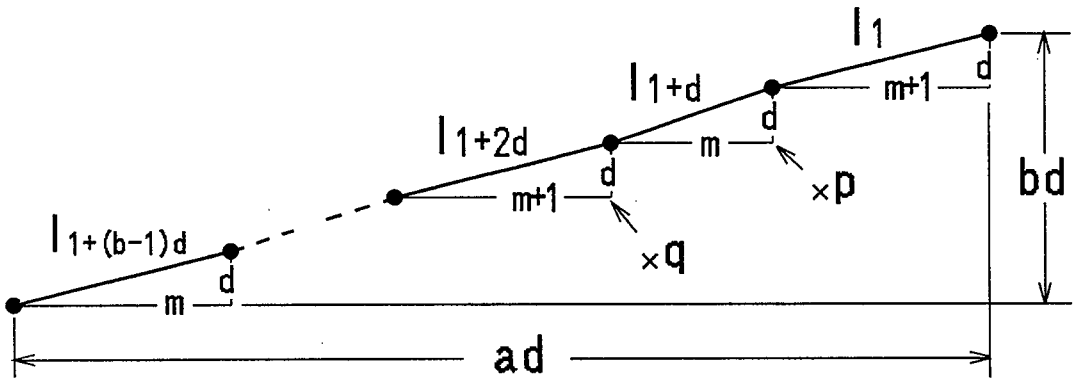


Fig.4.23 The kind and number constructing partial length chain

4．8 結言

本章の内容を要約すると，次のようである．

(1)2値化図形から輪郭線長を求める方法として SDLS 法を提案し，ソフトウエア処理に特に有効なこと，画像入力と計算処理が並列実行可能なこと，移動物体の実時間形状計測に適していることなどの特長を述べた．

(2)直線輪郭線について SDLS 法による測長値の理論式を導いた．

(3)正方形，円についてシミュレーションを行い，スキップの導入により，誤差，ばらつきが減少することを確認した．図形の輪郭が曲線の場合に測長精度が高いことがわかった．最適な s の値として s=1 を得た．また微小図形へ適用限界を明らかにした．

(4)測定誤差，ばらつき，処理速度，もしハードウエア化した場合の容易さなどの総合的な観点から，従来手法との比較を行い SDLS 法の優位性を明らかにした．

(5)直線エッジを間隔 d でスキャンしたときの部分長連鎖の種類と個数に関する定理を数学的に証明した．

本章にてディジタル測長に関する研究についての議論を終え，5章で応用例を挙げることになるが2章～4章で述べた内容を総括すると次のようなことが言える．

これらの章で測長手順の精度の検証に用いた例はディジタル測長の始端と終端が一致している矩形や円の輪郭長測定のケースを取り上げた．その理由は自由曲線を対象として輪郭長を測定するとすれば，提示した測長手順の測定精度が誤差1[%]以下になっており，実際上，真の長さの決定が困難で比較検証が不可能なためである．2章の最後で述べたように矩形図形の場合，1辺の長さL の直線部は面積に対してはL 画素分の，輪郭長については $(\sqrt{2}-1)\times L$ 分の誤差発生の可能性を内含しており，実験結果もこれを実証している．つまり測定のばらつきが大きかった．これに対して円弧の測長ではばらつきが非常に小さく，安定に結果が出ている．このことから先の自由曲線に対する測長値の安定性も推し量ることが可能である．実際，次章に挙げる応用例，例えば甘藷の形状判別の例では極めて明確に判別が可能となっている．

第５章　ディジタル測長の応用例

５．１　製品検用査への応用

３章で述べた周囲長測定ハードウェアは対象物体の投影面積等も同時に測定可能で，面積と周囲長から，当該物体の形状係数を算出できる．このことは対象物体の中の，基準から外れた，いわゆるいびつな物を摘出可能なことを意味している．この意味で応用の対象となる可能性のある物として農産物，建築材料，陶磁器製品等様々なものが挙げられる．隣接２線走査法はその最も大きな特長として１方向からの走査で，結果が実時間で求められる点にある．このことと先の各種製品が多くの場合，生産工程の中でベルトコンベアを用いて搬送されることを考慮するとテレビカメラのような２次元のイメージセンサ（エリアイメージセンサ）を用意する必要がない．つまり対象物体のほうが移動するので，撮像センサとしては１次元のイメージセンサ（ラインイメージセンサ）があれば十分である．

４章では上記の専用ハードウエアを用いず，従来の正方形画素としての画像入力装置をそのまま利用可能なように拡張したスキップ２線走査法（ＳＤＬＳ法）について述べた．この手法はソフトウエア処理に限定されるものではないがソフトウエア処理のフレキシビリティを利用して，ホスト計算機のディスプレイスクリーン上で様々な処理方法を指示できる点がその長所である．例えばポインティングデバイスを用いて図形輪郭上の任意の点間の輪郭長を求めたり，対象画面中のいくつかに分離された図形の中から，指示された部分に限定した輪郭測長あるいは面積の求積が可能である．後者の例は金属等を粉体化した場合の顕微鏡写真から個々の粒子を分離して面積測定する場合などである．勿論全体としての形状係数を測定すれば粉体の粒度の判定も可能である．

３章で示した隣接２線走査法（ＤＬＳ法）は専用ハードウエアを用いるから当然でもあるがその実時間性に特長がある．それに対して４章で示したスキップ２線走査法（ＳＤＬＳ法）はソフトウエア処理によるフレキシビリティに特長があるともいえる．一方，ここでこの両手法に共通する点を考えると対象物体の２次元投影図形を基に各種の処理を行うという点である．冒頭に述べた如くいくつかの応用の対象が考えられるがそれらに共通な特徴は平面的な物体であることである．この意味で例えば建築材料では瓦，タイルなどであり煉瓦，コンクリートブロックなど厚みのあるものはその一部が欠けていたりしても，１方向からの撮像では検出できる可能性が低いので撮像を 90° 違えた２方向から行う必要がある．しかしこれでも正確に欠陥抽出できるとは限らない．但し軸対称性を持つ物体の場合はこの限りではなく，例えば農産物の例として甘藷，いわゆるさつまいも，あるいはじゃがいも等が挙げられる．この場合は面積を用いて大きいもの，小さいものに、周囲長から計算される形状係数を用いて細長いもの，太くて丸いものに分類できる．

本章ではこれらに関する２つの応用例をあげる．一つはベルトコンベア上を移動する甘藷の選別装置である．対象は甘藷に限らず同程度の大きさの物体で軸対称性を持つものであれば選別に適用可能である．これについては５．２節で述べる．今一つは建築材料としてのコロニアル瓦の検査である．ここではソフトウエア処理のフレキシビリティを活かして指示された２点間の測長についても触れる．これについて５．３節で述べる．

5．2　ベルトコンベア上の物体の選別

先に述べた如く測定対象が一定速度で移動する場合には１次元のイメージセンサで撮像が可能である．こ種のセンサ（ＣＣＤイメージセンサ）は最近急速に普及しつつある家庭用ファクシミリや複写機などのために大量に生産され，価格も非常に安価になりつつある．またイメージセンサのクロックドライバなどの周辺回路は以前は外付けであったが，これらを集積回路として一体化したものが市販され，回路設計は極めて容易になった．パソコンをホストコンピュータとして，拡張スロットの仕様がわかっており，割り込みやＤＭＡの知識があれば容易に設計可能である．１次元のイメージセンサのもう一つのメリットはそれが非常に高分解能であることである．最低で１２８画素，大きいものでは８１９２画素のものまで市販されている．適用可能なクロックレートも急激に増大し，非常に高速に画像の取り込みが可能となった．このクロックレートの問題はベルトコンベアの移動速度と関連しており，撮像してできあがる，計算機あるいはハードウエア内のメモリイメージとしての画素寸法に大きく関係している．１ライン上の画素数を p［個］，クロックレートを c［MHz］，ベルトコンベアの移動速度を v［m/s］，撮像幅を w［m］，画素の縦横比を rとするとその間には次の関係がなければならない．ベルトの進行方向を画素の縦方向，これに垂直方向を画素の横方向とすれば

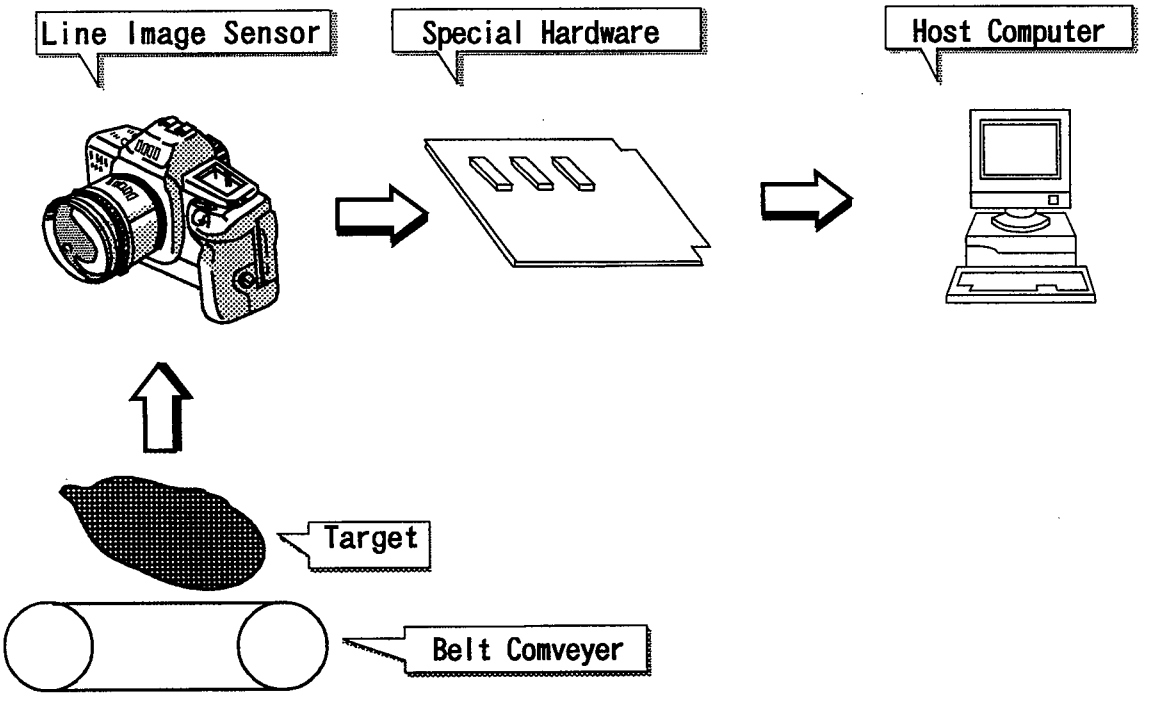


Fig.5.1 System configuration

画素横寸法＝ w/p［m］
一回のスキャンに要する時間＝ $p/c\times10^{-6}$[s]
この時のベルトの移動量＝画素縦寸法＝ $pv/c\times10^{-6}$[m]
縦横比 r ＝画素縦寸法／画素横寸法＝ $vp^2/cw\times10^{-6}$[ϕ]
縦横比 r を先に決めればベルトの速度 v ＝ $rcw/p^2\times10^{6}$[m/s]

本節ではこれらの知見をもとに対象を甘藷としてその選別装置を試作し，実用装置としての問題点の抽出を行った．先に述べたように対象は軸対象なもの，あるいは平面状の物体であればよいので他への応用も十分可能である．

５．２．１　システム構成

図５．１に本試作装置のブロック図を示す．隣接２線走査法のアルゴリズムはハードウエアで構成した．この部分はホストとなるパソコンの拡張スロットから汎用拡張ボードを経由して接続されている．ＣＣＤイメージセンサを含む撮像部は市販のカメラを改造して製作した．そのフィルム面と同じ位置にセンサが取り付けられている．製品はＳＯＮＹ製ILX503である．分解能2048画素のものを1024画素に落として使用した．ホストコンピュータはＮＥＣ製PC-9801DS（i80386SX 16[MHz]）である．ベルトコンベアの駆動源はかなり大型のステップモータである．この部分は最初，単相の誘導電動機を用いていたがベルトの速度変動が大きいことからステップモータに変更した．速度変動は直接的に面積，形状係数の測定値に影響を与え，実際にそのばらつきが非常に大きいことが判明したからである．

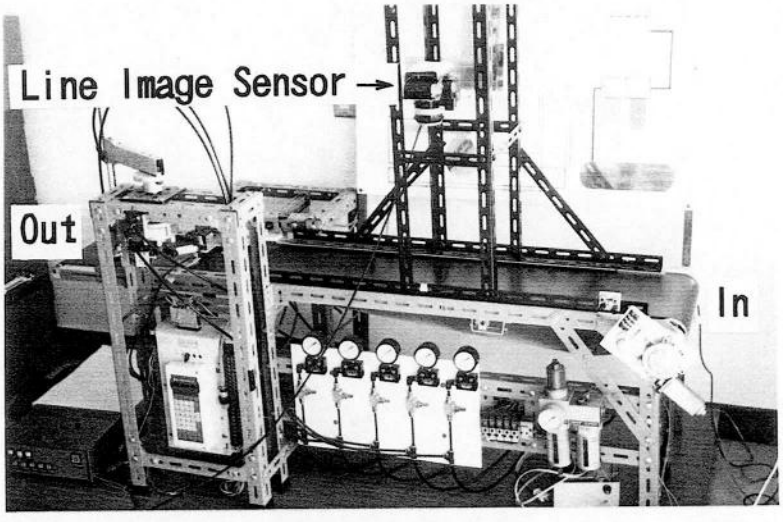


Photo.5.1 Experimental device

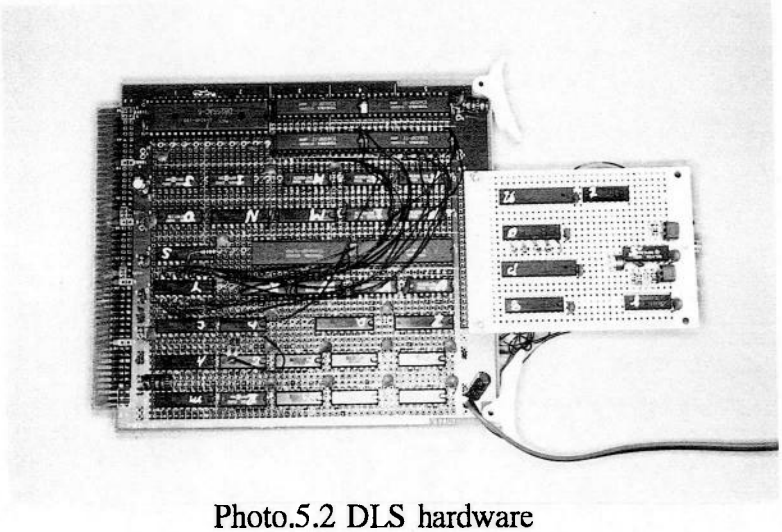


Photo.5.2 DLS hardware

写真５．１は装置全体を示している．イメージセンサはアングルで組んだ支持部から下方向に向けてセットされている．写真５．２は隣接２線走査法を組み込んだ専用ハードウエアである．この部分の回路構成は３章の図３．１５とほぼ同じである．異なる点は垂直同期信号が物体の通過にあわせて作られる点と，１走査線分のクロック数が１０２４（実際には画像の無信号部を含めるのでこれより若干多い）に変更されていることである．また対象物の通過後専用ハードウエアから割り込み信号が入り，ホストコンピュータは専用ハードウエアからの処理結果を受け取る．ベルトコンベアの出側には空気圧制御のマニュピレータを設け，プログラマブルコントローラ（シーケンサ）を通してホストコンピュータからの指示により，選別振り分けされる．周囲長と面積，従って形状係数を測定できるようにし，重心の測定は行わなかった．設定したパラメータは次のとうりである．

画素の縦横比 r =2.5　　---これは３．３．４項の議論から決まった値である．
画素数 p =1024[個]
クロックレート c = 1 [MHz]
撮像幅 w =0.3[m]
ベルトコンベアの移動速度 v =0.71[m/s]

（a）　　（b）

Photo 5.3 Testing Samples

５．２．２　実験結果

試作装置の測定系の校正は次のように行った．真円の紙によるデータサンプルを作り，この形状係数をできるだけ１に近くなるようベルトコンベアの速度を微調整した．結果的には形状係数を1.005，ばらつきの標準偏差 σ =0.002とした．写真５，３は実験に使用した２種の甘藷のサンプルである．人間の目で手作業で選別するには容易な有意差がある．左(a)は細長いもの，右(b)は太くて丸いものである．重量はほぼ同じで(a)251[g]，(b)248[g]である．サンプルの甘藷は完全な軸対称性はない．従って同じ甘藷を複数回測定すれば姿勢が異なるので異なったデータが出る．従って多数回測定してその平均値とばらつきを算出すべきであるが甘藷の表面の凹凸により姿勢はおのずと決まってしまう．（安定位

置が少ないの意）そこで撮像に影響がないような支持物を置いて中心軸と直角に４種の姿勢で実験を行った．このデータを表５．１に示す．

Table 5.1 Measured data for two samples

Position	Sample (a) Thin			Sample (b) Fat		
	Area [Pixel]	Perimeter [Pixel]	Figure factor	Area [Pixel]	Perimeter [Pixel]	Figure factor
1	176828	1938.3	1.69	139364	1418.0	1.15
2	171024	1998.2	1.86	138092	1437.0	1.17
3	177204	1940.2	1.69	135280	1417.2	1.18
4	173916	2017.1	1.86	136420	1419.9	1.18
Mean	174743	1973.4	1.78	137289	1422.8	1.17
σ	2495	34.8	0.08	1561	8.3	0.01

５．２．３　考察

（１）装置の校正

試作装置の校正の際，形状係数を真円に対して最終的に1.005とした．第３章の表３．３でも若干の正の誤差（形状係数=1.013）が出ているのでこの誤差はアルゴリズムに起因するものと考えられる．コンベアの速度調整で１に近ずけるのは本質的なやり方ではない．というのは速度の変化は画素形状の変化をも伴うからで（従って面積測定値も影響を受ける），真円時の値１に無理に一致させることはせず微調整にとどめた．長方形，３角形等を対象とした場合，理論値と一致するという保証がないからである．

（２）選別精度，速度

表５．１のハッチング部に形状係数の平均値とばらつきσを示しているがサンプル(a),(b)では明確な差が出ており，実験の目的は十分達成されたと判断できる．作業能率を上げるためコンベアスピードを上げるとr一定ならクロックレートを上げなければならないが現在市販のラインイメージセンサで十分対応可能である．実際この種の実験を試みたが試作装置のベルトの長さが短く，入側にサンプルを置いても甘藷の場合は転がるので，センサの下を通過するまでにサンプルの姿勢が安定せず，正確なデータは取れなかった．

（３）甘藷を選別する事の意義

甘藷は市中の販売店ではグラム単位の袋詰めで売られている．この段階では既に大きさ，形の選別が成されているものが多い．現にこの実験に用いたサンプルを購入するのに２袋買うことを余儀なくされた．いわゆる”やきいも屋さん”の話では，太くて丸いものは焼くのに時間がかかる．均一に焼けないので嫌われるとのことである．卸売り市場でこの選別したものとそうでないものには多少の価格差がある．出荷する農家は選別を行っていくらかの競り値のアップを期待する者とそうでない者とまちまちであると聞いている．

５．３　建築材料の検査

本節では第４章で述べたディジタル測長法の応用例として、近年，住宅の屋根ふき素材として多く採用されるようになったコロニアル瓦を対象として、平面図形の形状パラメータを測定する例を挙げる．この瓦は素材原料は石綿でこれを厚さ9[mm]程度の板状に整形

し、防水加工を施したものである．形状は様々なものがあるがここでは形状の一部に自由曲線を含んだタイプのものを用いた．その自由曲線で構成される図形の一部分をポインティングデバイスにより始点、終点を指定し測長する例を示す．そして形状の一部に欠けた部分をもつ欠陥のある瓦と正常な瓦との比較検討を行う．

５．３．１　システム構成

図５．２に本システムの構成図を示す．対象となる原画像はモノクロＣＣＤカメラで（570×485画素）撮影した．これからの映像信号はＴＶモニタ(Moniter 1)に入ると同時にＡＤ変換されて（512×504画素、分解能8Bit）計算機メモリ内に取り込まれる．このデータは専用のビデオRAM部に格納され，逆にＤ／Ａ変換されモニタ(Moniter 2)に表示される．これはビデオプリンタ(Video Printer)にも接続されている．この画像はホストコンピュータ(Host Computer)付属のディスプレイにも同時に出力できる．ただしこのディスプレイの解像度は512×400、グレイスケールは16レベルとかなり画像品質が劣る．しかし２値化画像を取り扱う上ではこの問題はない．測長の前処理としてまず画像データを２値化してモニタ(Moniter 2)上に出力した．写真５．４はモニタ画面のハードコピーである．図中左は正常な製品、右は欠陥品である．以下同様の順序で示す．次にこれをグレイスケ

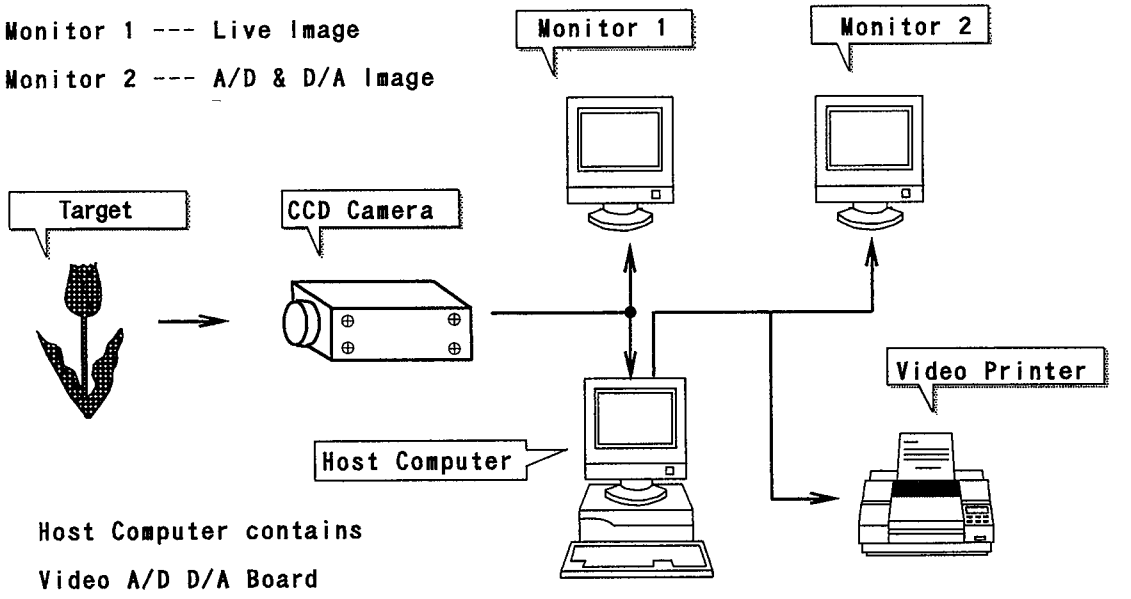


Fig.5.2　System configuration

ールの丁度中心(128)を閾値として２値化した．写真５．５にこれのハードコピーを示す．次に２次元微分処理を行いエッジを抽出した．微分にはもっともシンプルなRobertのオペレータを使用した．写真５．６はこのハードコピーである．

計算機付属のディスプレイの方はポインティングデバイスとしてのマウスと連動して図形内の位置を指し示すことが可能なので前章のＳＤＬＳ法と組み合わせて図形輪郭上の任意の２点間の測長を行うプログラムを作成した．この様子を写真５．７に示す．同図内の２個の黒丸が始点と終点を表す．

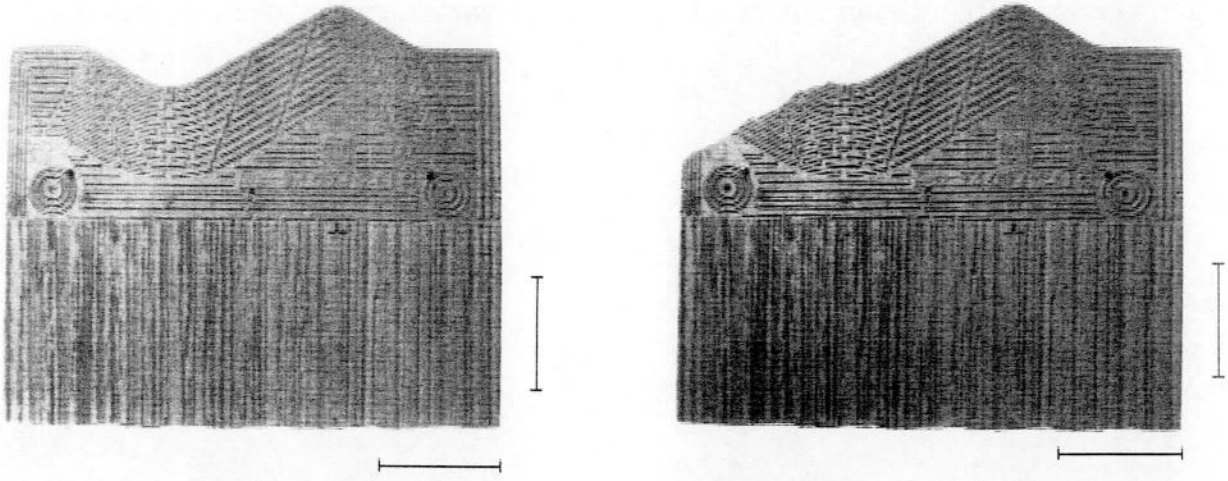


Photo.5.4 Original image (left – normal , right – with defect)

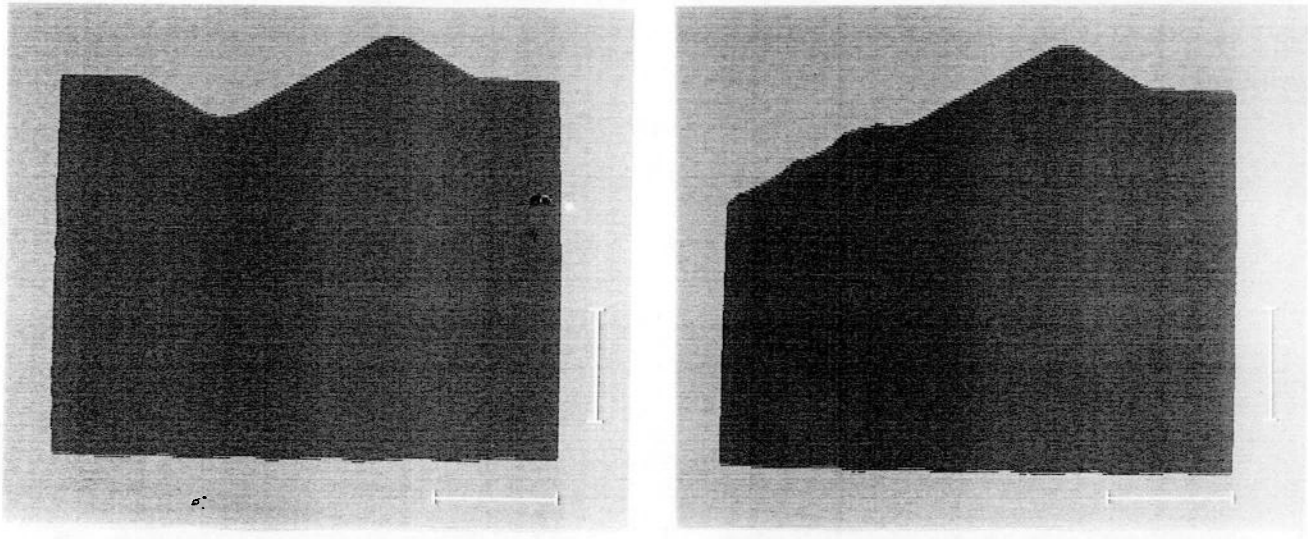


Photo.5.5 Converted to binary image (left – normal , right – with defect)

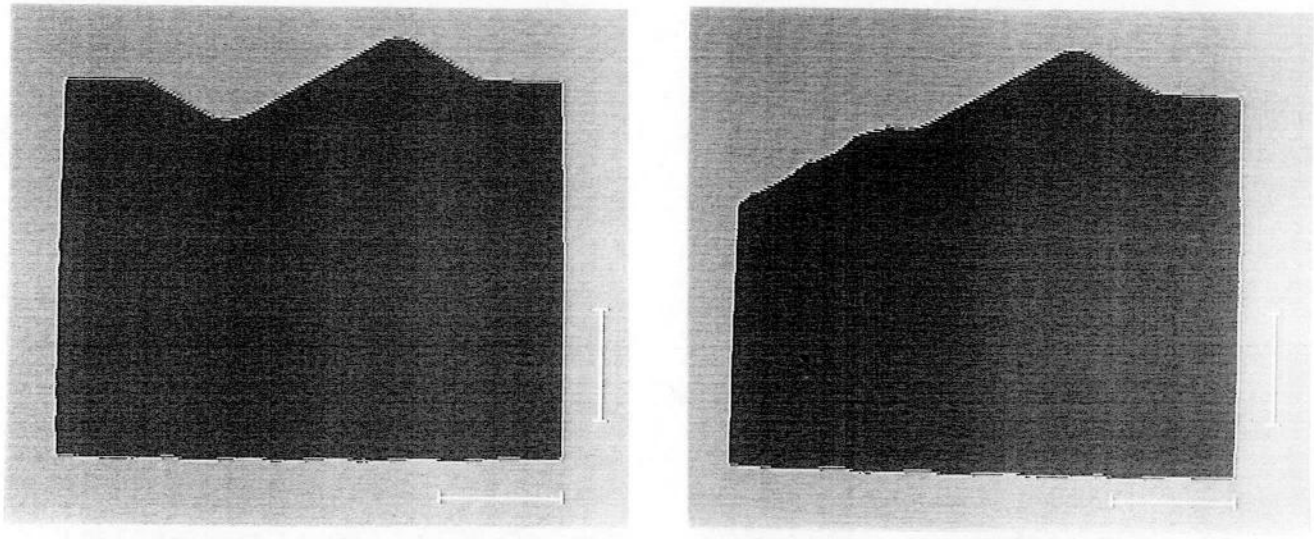


Photo.5.6 Contour is extracted (left – normal , right – with defect)

Photo.5.6　Contour is extracted　(left − normal , right − with defect)

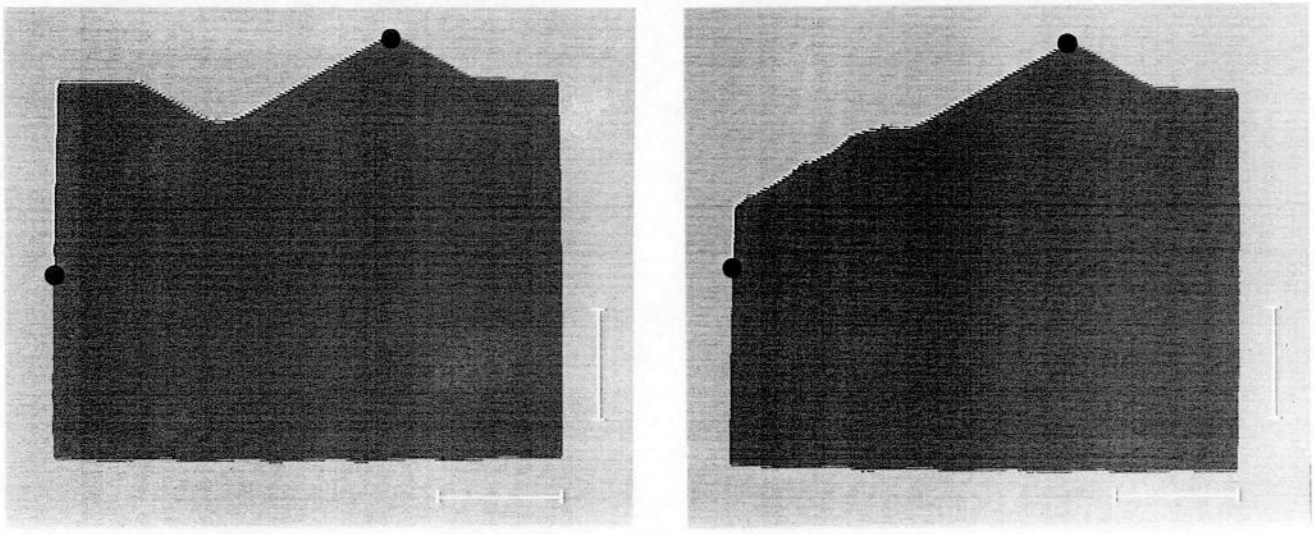


Photo.5,7 Directed contour length is measured　(left − normal , right − with defect)

各図中，右下の縦横の線は寸法比較用のスケールでそれぞれ100[Pixel]に相当する．

図５．３は輪郭長測定プログラムの手順を説明する図である．始点(a)、終点(b)と測長の方向を示す(c)点の３点をポインティングデバイスで入力する必要がある．前記写真５．７の場合始点から反時計回りに測長を指示した．始点と方向が指示されるとプログラムはY方向に±5[Pixel]，X方向に±5[Pixel]の範囲で輪郭を探索しながら測長アルゴリズム

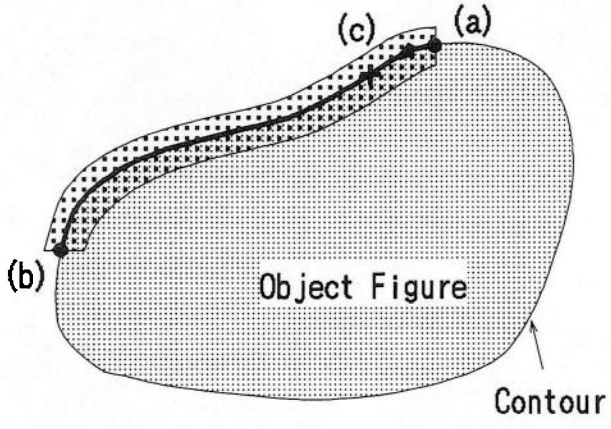


Fig.5.3 Procedure of digital length measuring between directed two point

を適用して測定を続ける．図５．３の粗いハッチング部分はこのアルゴリズムの適用範囲を模式的に描いたものである．このように輪郭に沿った狭い範囲を測長するので測定時間が短縮される．

５．３．２　実験結果

第４章に示した手順で，他の図形パラメータ測定プログラムも使用して面積，周囲長，指示された区間の輪郭長のデータを採っている．これらを実測値（対象物を物理的に計測

した結果）と比較して次にまとめて記す．最左欄の項目で面積は写真５．５の黒い部分の面積である.周囲長は写真５．６の輪郭（白い部分）をＳＤＬＳ法により求めた.

Table 5.2 Parameters of object figures

	Obtained by measuring program		True value	
	Normal	Defect	Normal	Defect
Area Photo.5.2	288282[mm²]	278126[mm²]	290588[mm²] Error 0.8[%]	280629[mm²] Error 0.9[%]
Perimeter length Photo.5.6	2239[mm]	2139[mm]	2272[mm] Error 1.5[%]	2167[mm] Error 1.3[%]
Contour length Photo.5.7	691[mm]	581[mm]	705[mm] Error 2.1[%]	591[mm] Error 1.7[%]
Figure factor	1.384[ ]	1.309[ ]	1.413[ ] Error 2.1[%]	1.332[ ] Error 1.8[%]

３番目の指定された２区間の輪郭長は同じくＳＤＬＳ法により前述の方式で求めた.
右欄に示す真値（実測値）は次のようにして得たものである.面積は瓦の実物を網目で区切ってその数を数えた.周囲長は同じく実物の周囲に沿って糸をめぐらしその全長から求めた.輪郭長も同じである．１画素あたりの実寸はX方向 1.572[mm/Pixel],Y方向1.570[mm/Pixel]であった．第３章で述べたように光学系の歪みが測長精度に大きく影響を与えることが判っているのでこの実験に用いたＣＣＤカメラは高精度の光学系を持つ別のカメラ（2/3インチ工業計測用ビデオカメラ DXC-151A ソニー製）を用いた．その結果全体としてかなり精度の高い測定が出来た.

５．３．３　考察

（１）誤差評価

面積の真値はメッシュサイズの一定な金網を対象物の上に重ねて升目を数えて測定したが輪郭が升目の中間にかかるとき"1"/"0"の判定が困難で真値自体にも多少のばらつきがある．表での誤差0.8[%]（正常な瓦），0.9[%]（欠けた瓦）という値は非常に精度が高いと考えて良い.

周囲長及び指定された輪郭長についても同様なことが言えて糸を延ばして全長を測定する際の延び縮みが真値の測定誤差に影響する．従ってこれにもばらつきがある．表では約2[%]に収まっているがこれも十分な精度と言えるだろう.

（２）処理速度

指定輪郭長を求めるに要した時間は画面全体にＳＤＬＳ法を適用した場合（写真５．６）と比較して時間にして1/15程度であった．写真５．７においてＳＤＬＳ法を適用する面積は図形輪郭に沿った±５[Pixel]の範囲であり画面全体に適用する場合（写真５．６）に比し1/15よりはるかに少ない．単純に考えればこの面積比に比例した処理時間となるべきであるが輪郭追跡の手順の実行時間が加えられこのような値になったものと思われる.

第６章Hough変換過程の分析と新手法の考え方

本章ではHoughによって提案された直線検出手法について，その後様々な機能が発見、考案されてきた経過を振り返りながら次の第７章，８章，９章で述べる筆者の新しい提案、工夫についてその概要を述べる．

６．１　Hough変換による直線検出の原理

Hough変換の原式は

$$\rho = x \cdot \cos\theta + y \cdot \sin\theta \qquad \text{---(6.1)}$$

である．まずX-Y平面上で与えられた点P(x, y)に対して式(6.1)の各変数は図６．１（ａ）に示された関係を満足する．すなわちρは原点から点Pへ向かうベクトルの長さでありθはX軸と成す角度である．ここでベクトルρをこれと垂直な直線Lに対応付けしておく。次に式(6.1)を満足する別のθ'，ρ'を考え，図６．１（ｂ）を参照することにより（θ'，ρ'）の組は点P(x, y)を通りベクトルρ'と垂直な直線L'に対応付けされることになる．

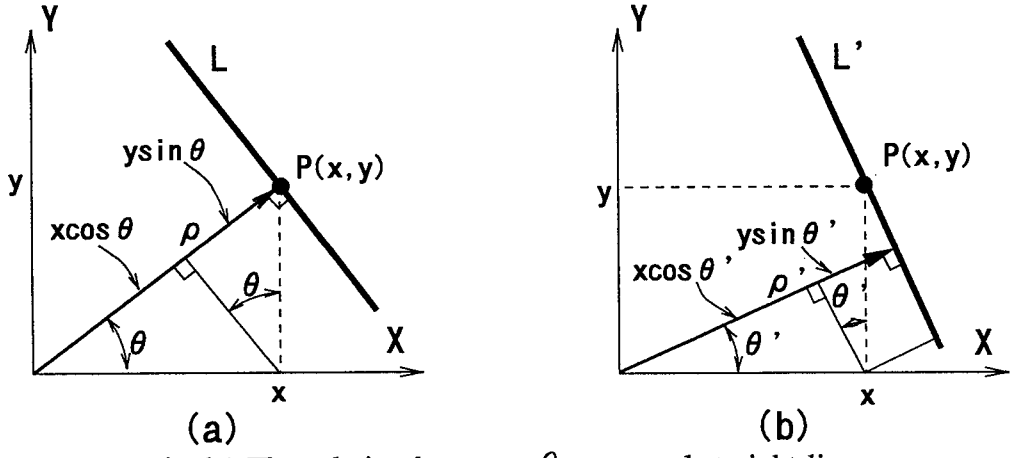


Fig.6.1 The relation between θ , ρ and straight line

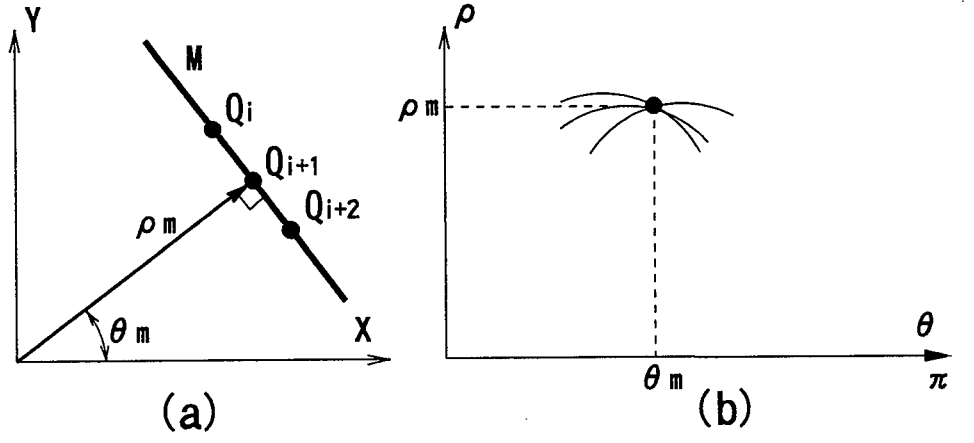


Fig.6.2　The relation between θ_m, ρ_m and line M

すなわち式(6.1)でθを変化させて求めたθ'_i，ρ'_i($0 \leqq i < n$)の組み合わせ（θ'_i，ρ'_i）は点Pを通りベクトルρ'_iと垂直に交わるn個の直線L'_iと対応付けされるわけである．こ

こでθを横軸ρを縦軸とする直交座標系に（θ'_i, ρ'_i）をプロットすれば式(6.1)の形から正弦曲線を描くことがわかる．ここで図６．２（ａ）の如く直線M上の点Q_i($0 \leqq i < n$)を与えて計算すると$\theta-\rho$平面上にn個の正弦曲線が描かれるがこれらは共通に(θ_m, ρ_m)を通過するはずである（図６．２(ｂ)参照）．ここで(θ_m, ρ_m)は最初に与えた点Q_iを含むX,Y平面上の直線Mに垂直なベクトルρ_mとX軸のなす角度θ_m及びρ_mの大きさρ_mである．以上のことから次のことが分かる．

まずX-Y平面上の任意の点群（これをサンプリング点と呼ぶ）に対して式(6.1)により$\theta-\rho$平面上に写像変換を行って，$\theta-\rho$平面上でのその正弦曲線の通過度数を累積する．その後$\theta-\rho$平面の各点について曲線の通過度数のピーク点を検出しそれを(θ_m, ρ_m)とすればMなる直線が検出されるわけである．直線の式は式(6.1)をyについて解いて

$$y = -\cot\theta_m \cdot x + \frac{\rho_m}{\sin\theta_m} \qquad \text{---(6.2)}$$

これがHough変換による直線検出の原理である．その後の研究によって$\theta-\rho$平面上の正弦曲線の分布を解析することにより，円，楕円が検出でき，X方向フェレ長，Y方向フェレ長，絶対最大長，凸包図形の絶対最大長，周囲長などを求めることが出来るようになった．

以上の手順からわかるように原画像上の直線は，推定できるある程度の長さを持っておればよく，ノイズなどで直線がとぎれていても検出可能である．これがHough変換による直線検出法の大きな特長である．

最初に与えたX-Y平面上の点，すなわち**サンプリング点**はHough変換によって検出しようとする目的の図形パラメータに直接関連付けされたX-Y平面上の座標である．X-Y平面上のサンプリング点P(x, y)が与えられたときθをn個に分割しθ_i($0 \leqq i < n$)について式(6.1)のρを計算する過程は正弦関数を含んだ計算をn回行うことになる．この計算を**Hough変換演算**と言う．サンプリング点の数は通常の画像の場合数百～数千個になりその計算コストは非常に大である．さて$\theta-\rho$平面上の曲線の通過度数はディジタル信号処理では２次元配列の各要素に累積することになり，いわば２次元の度数分布表である．

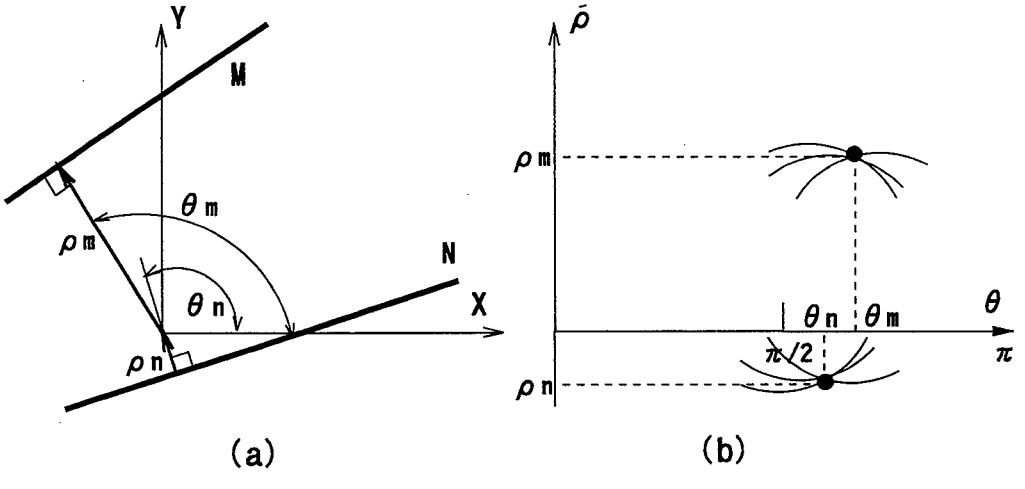


Fig.6.3 Position of detected line when ρ <0

この意味でこの２次元配列を**θ-ρヒストグラム平面**と呼び度数累積の過程を**θ-ρヒストグラムの形成**と言う．またθ-ρヒストグラム累積度数のピークを探す過程を**θ-ρヒストグラムの探索**と呼ぶ．以上のようにしてθ-ρヒストグラムの探索を行うことによって度数のピーク点から原画像中の直線を検出可能なことがわかった．ここで図６．３のようなケースについて考察しておく．（ｂ）図で検出された(θ_m, ρ_m)（いずれも正の値）は（ａ）図で直線Mに対応することは明らかである．しかし検出されたもう一つのピーク点(θ_n, ρ_n)ではρ_nが負である．この場合ベクトルρ_nの向きは（ａ）図の原点から直線の方へ向かうのではなく丁度その逆になっていることに注意しなければならない．この場合検出される直線の位置は（ａ）図のNである．

６．２　Hough変換の各過程

前節で述べたようにHough変換を用いればノイズが混じった低品質の原画像から直線の位置の特定が可能である．逆にHough変換では一般的にエッジの抽出に微分オペレータを適用するので画像濃度の異なる部分で本来はエッジでない部分が疑似エッジとして現れたりノイズが強調されることがある．このようにHough変換による直線検出にはまだ手作業の介入を必要とする場合が多い．第１１章で述べる応用例は画像の濃度比が非常に大きく姿勢が安定しているので自動化可能となったものである．

さてHough変換のための原画像は一旦２値化された後，エッジの抽出が行われる．またエッジ上のサンプリング点の数そのものを低減させる試みも見受けられる[68]．エッジ抽出の手順の中で，ある程度のエッジの連続性に関する情報を得ておくことも考慮の必要があると考えられる．

第１章でも述べたが更にHough変換による直線検出の過程を詳細に分類して図で表現したものが図６．４である．

①原画像は撮像装置によって捉えられ計算機のメモリイメージとなる．
②原画像を２値化する．
③２次元空間微分によってエッジを抽出する．あるいは目的に応じたサンプリング点を取り出す．
④Hough変換原式(6.1)によってθに対応するρ値を計算する．--Hough変換演算
⑤θ-ρヒストグラム平面への度数累積を行う．----------**θ-ρヒストグラムの形成**
⑥θ-ρヒストグラム平面の度数探索を行い
　　直線の位置を決定する．----------**θ-ρヒストグラムの探索**

①，②，③はHough変換の前処理と呼ばれておりこの手法の本質的なものではない．以下この①～③について簡単に触れる．④～⑥についてはHough変換の中核を成す大きな３つの過程なので第７章、第８章で述べる内容に関連させ，その問題点と解決法を含めて６．３節～６．５節で概観する．以下，これら３つの過程は上記ゴチック体で記した言葉で短く省略して引用することがある．６．６節では④Hough変換演算の高速化について新手法の基本的考え方を述べる．

（１）撮像装置

前処理過程①の撮像装置としては工業計測用カメラが一般的であるが，次のような場合

には１次元イメージセンサが用いられる．すなわち撮像装置が固定されており測定対象物が移動する場合である．つまりベルトコンベアで搬送される物体などを捉えるケースである．第１０，１１章でこのような応用例を挙げるがこの場合コンベアの移動速度の安定性が非常に重要になる．

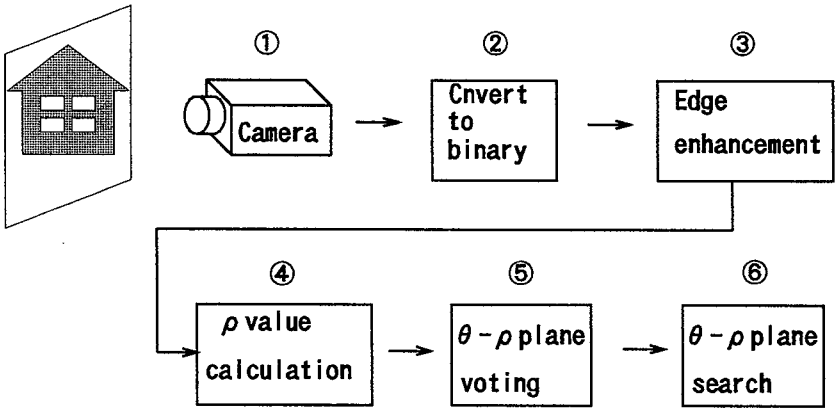


Fig.6.4 Procedure of Hough transformation

その理由はコンベアの移動速度が撮像される画像の移動方向分解能に直接関係するからである．筆者が経験したいくつかの装置ではこの移動速度のばらつきが標準偏差σで表して0.1<σ<5%程度ばらついていた．面積、周囲長などの測定精度が１％近くになるとこの点が問題になる．σ＝５％の装置はコンベアの駆動負荷に比し駆動源であるモータの出力が小さいものであった．もちろんコンベアの１回転中の摩擦負荷の変動が大きいことも原因の一つである．このような装置に用いられるモータは単相、３相の誘導モータの場合がほとんどで負荷の変動がスリップsの大きな変動となり移動速度に変化を生じる．モータの出力に余裕があるときはスリップの変動も小さいから安定する．

第１１章で述べる応用装置のプロトタイプとしての実験装置ではかなり大型のステップモータを使った．この場合ベルトのスピードは供給されるパルスレートにのみ依存するのでばらつきはほとんど０である．また第３章で述べた光学系のひずみの問題も重要である．第１１章の応用例ではこの点については光学系に精度の高いものを用いたこと及び１次元イメージセンサを採用したからかもしれないが大きな誤差は見受けられなかった．

（２）原画像の２値化

前処理過程②画像の２値化に関しては周囲の濃度との関連で閾値をダイナミックに変化させながら処理する方法も提案されている[2]．基本的には濃度ヒストグラムを用いて頻度の谷部を選ぶ方法がよく採用されるが谷部が判然としない例も数多い．また応用の用途によってはその閾値設定値がある程度領域毎に指定できる場合がある．また自動車のナンバープレートの検出への応用例[27]では抽出領域が限定できるので他の部分をマスクしてしまうことも考慮されている

（３）画像の２次元空間微分

最後に前処理過程③の２次元空間微分について述べる．第６．１節で記したように２次元空間微分の手順は原画像の２値化と前後することもある．微分オペレータとしては注目

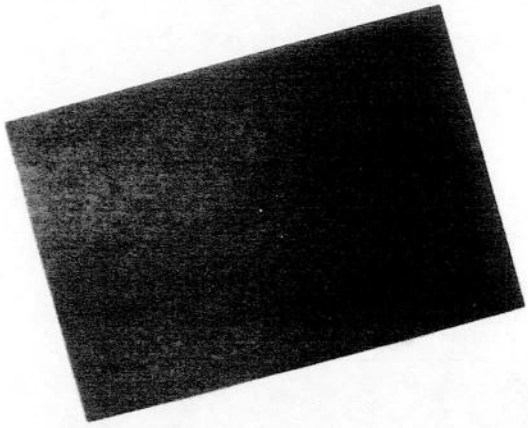


Photo.6.1 Original Image

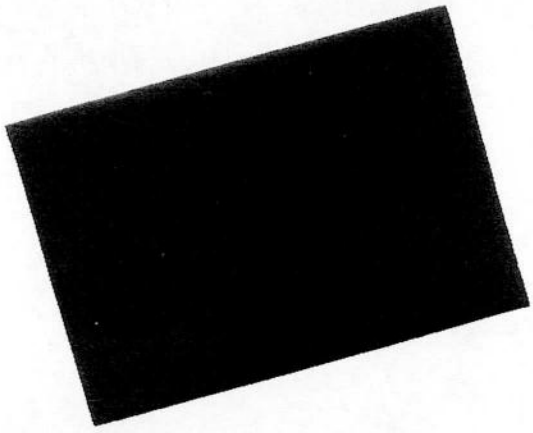


Photo.6.2 Convert to binary image

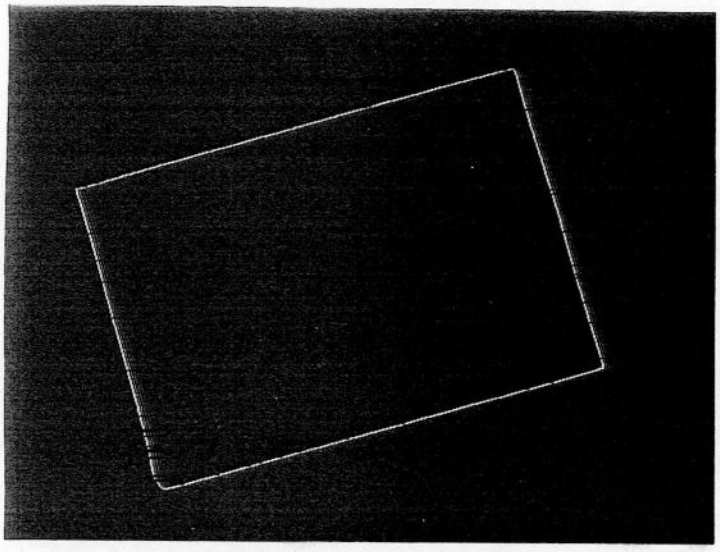


Photo.6.3 Edge is enhanced

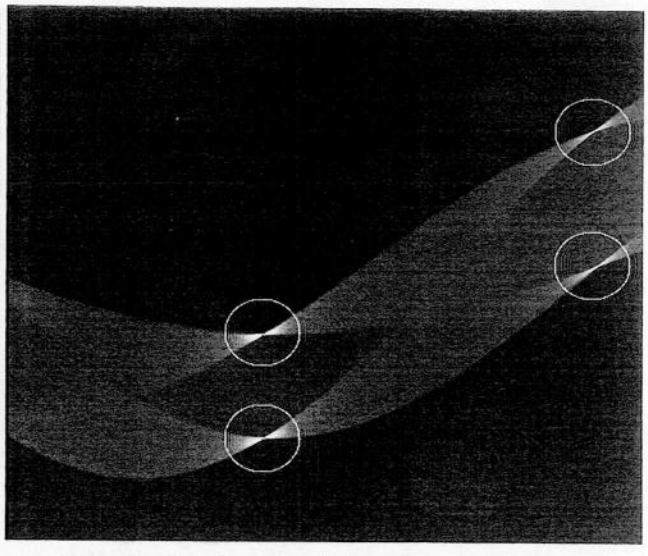


Photo.6.4 Hough plot

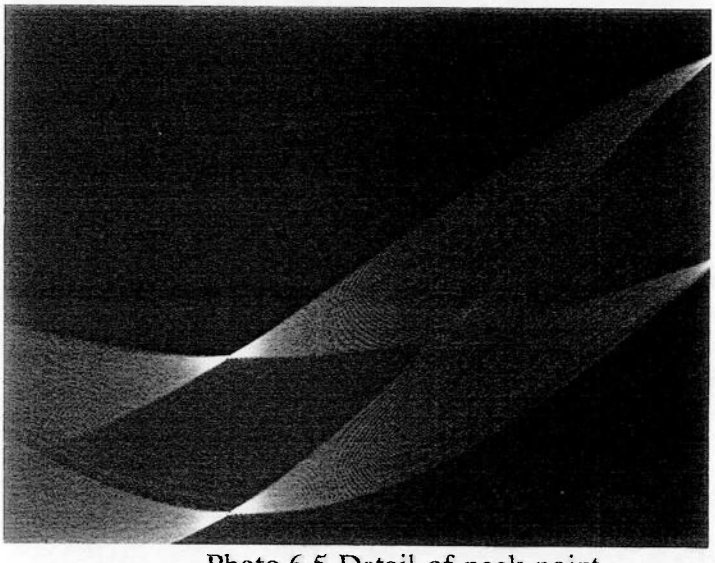


Photo.6.5 Detail of peek point

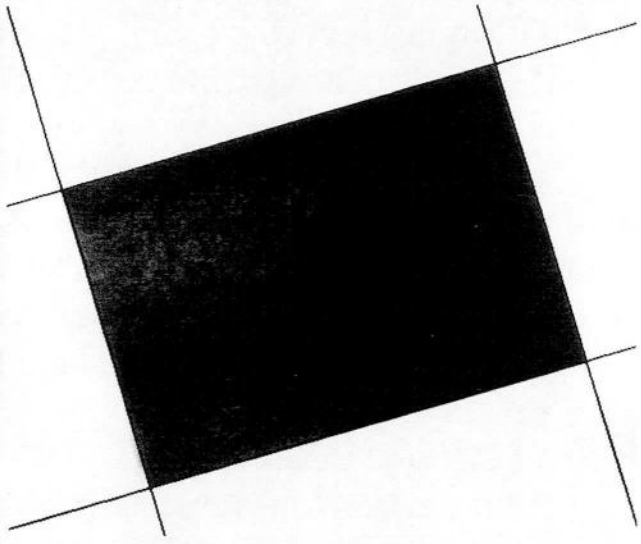


Photo.6.6 Detected line is overlaid

している画素のまわりの８連結画素をもとに計算するラプラスのオペレータ、注目している画素近傍の４画素をもとに計算する方法(Robertのオペレータ)などがある．前者は専用のＬＳＩが市販されている．

一般的に微分操作は画像中のノイズを強調するきらいがあるので微分の重みづけをあまり大きくすることは好ましくない．本論文では手順の簡素化を含めて全て,画像の２値化を先に行い、後者の方法を適用している．

（４）Hough変換の例

写真６．１～写真６．６はHough変換処理による直線検出の過程を示したものである．前節で述べ，図６．４に示した手順の番号と関連付けて説明すると次のようになる

①---写真６．１は入力となる原画像である．白を背景として長方形の黒い目標物体が写っている．

②---これをある閾値で２値化したものが写真６．２である．

③---２値化イメージから２次元空間微分によってエッジを抽出したのが写真６．３である．

④⑤---Hough変換演算を行い，θ-ρヒストグラムを形成した結果が写真６．４である．白丸で囲んだ部分が度数ピークの存在箇所である．ピークの所で切れているように見えるが拡大すると写真６．５のようになっており切れてはいない．

⑥--θ-ρヒストグラムを探索し原画像上に検出された直線を重畳したのが写真６．６である．なお写真の座標原点はいずれも左上隅で、Y軸、ρ軸の正方向は下向きとなっている．（第１０．４節，図１０．７参照参照）

６．３　Hough変換演算

この過程はHough変換の実行過程でもっとも時間またはメモリ容量を必要とする部分である．そのために多くの研究がなされている．第１章でこれまでの成果を概括しているがこの問題は本論文の主たるテーマの一つであるから本節では，演算時間，必要とするメモリサイズなどの観点から問題点について議論する．なおこの演算の高速化については６．６節で筆者が新しい提案する手法について論じている．個々の手法の詳細については第７章、８章で述べている．

Hough変換演算は与えられた(x,y)について式(6.1)によりρ値を求めることに尽きる．式(6.1)を詳細に見ると$\cos\theta$，$\sin\theta$の求値，これとxまたはyとの乗算と両者の加算である．一般的な計算機の高級言語では正弦関数の精度は単精度浮動小数点数の場合，１０進約６桁である．この求値にはミニマックス近似の関数近似が多く採用されており，定数６～７個を含んだ積和演算で求めている．浮動小数点の乗算６回加算６回ほどである．これが$\sin$と$\cos$の２回と先の乗算２回と加算１回を加えて合計浮動小数点数の乗算１４回，加算１３回の演算である．計算機がハードウェアによる浮動小数点コプロセッサを標準的に内蔵するようになって加減乗除算はかなり高速化され加減算と乗除算の実行時間差はほとんどなくなった．しかし１個のρを求めるのに上記の回数の演算は大きな負担である．標準的なθ軸の分割数は512程度であり，演算に供されるサンプリング点の数が3000程度だとして１画面分のHough変換演算に必要な浮動小数点演算の回数は(14+13)×512×3000=41

472000回となる．実時間処理を考え，その基準をNTSC方式映像信号の１フレーム転送時間1/30[s]と考えると1/30/41472000=8.04×10^{-9}[s]となる．つまり約8[ns]となる．最近ではパソコンと呼ばれる計算機でもクロックレート200[MHz]（クロック周期5[ns]）程度のマシンも安価に入手可能になったが乗算を行うには数クロックから十数クロックを必要とし十倍以上の開きがある．また画像取り込み時のDMAに必要な時間、θ-ρヒストグラムの形成、探索に必要な時間も考慮する必要がある．Hough変換がリアルタイムの実用装置に組み込まれるとき，上の条件を満足するような高速の（大型の）ハードウエアリソースを利用することは費用の点で難がある．そこで何らかの高速Hough変換演算方法が必要になる訳である．その大きな流れとして表参照方式、連立漸化式を用いる方法がある．前者は$cos\theta$，$sin\theta$の計算に含まれるかなりの数の乗算、加算をなくしてしまうものである．つまり$cos\theta$，$sin\theta$の値を変化する各θについて予め計算して配列メモリに格納しておき，その値を配列要素の参照により取り出す方式である．

式(6.1)を見ると右辺にはx, y, θの３つのパラメータがある．もし右辺から乗算，加算を全てなくし表参照のみで演算しようとすれば３次元の表が必要となる．この点について塩野の報告[25]によれば予想したほどには速度が向上しないという報告もある（理由は後述）．$cos\theta$，$sin\theta$のθに関する項の表を用意した場合，必要メモリ量は無視できるほどであるが乗算２回と加算１回が必要である．このように演算速度とメモリ使用量はトレードオフの関係にある．式(6.1)を見ての単純な発想では効果的な方法は見つからない．６．６．１項，６．６．２項でこのような表を参照して$cos\theta$，$sin\theta$の値を求値する場合の表の構成法について概説する．第７章では２種類の表の構成法について詳述している．

次に連立漸化式を用いる方法について述べる．これはサンプリング点P(x, y)を初期値として，逐次的にρの値を求めていくもので使用するメモリ量も少なく非常に効果的な方法である．演算に関しては乗算をシフト算に置き換えており，速度も速い．シフト算は計算機の持つ命令群の中で少ないクロック数で実行できるものの一つである．また連立漸化式であるからこれをハードウェアに組み込めばρ値がクロック１個につき２個或いは４個並列的に求めることが可能となる．以上の方法について６．６．３項で従来からあるこの種の手法について概説する．第８章で筆者が提案する新しい計算（演算）方法についてその詳細を示す．

６．４　θ-ρヒストグラムの形成

前節に述べたようにρ値の計算にはかなり工夫の余地がある．そこで相対的に問題となるのが$\theta-\rho$ヒストグラムの形成に要する時間である．純粋に計算機のみを用いる場合，この過程は正弦曲線が通過する点に対応する**２次元メモリの読み出し，加算（＋１），再書き込み**である．最近のＣＰＵはパソコンに搭載される程度のものでも１次キャッシュ，２次キャッシュを備えている．θ-ρヒストグラム平面の大きさを512×512程度にとってもその大きさは256[Kb]程度でキャッシュが有効なのは確かである．しかしそれは読み出しのときであって，再書き込みの際にはメインメモリへの書き戻しが必要で速度が思うように上がらない．メインメモリは全てダイナミックラムが使われており，ＩＣ１個当たりの容量は16[Mbit]程度になりつつあるが，そのアクセス時間は50[ns]を切ることは当分な

いであろう．従って何らかの従来方式から発想の転換をはかったデバイスが考えられねばならない．その意味で第９章で示す電荷蓄積素子を用いる方法は画期的なものである．第９章では本節の問題を解決する方法として２次元ＣＣＤイメージセンサを電荷蓄積素子として利用した新しい手法を提案している．原理の概要は次のとうりである．後章で述べるどの方法でもかまわないがいずれかの方法でρ値を発生させる．このディジタル量としてのρ値と，対応するθ値をＤ／Ａ変換してアナログ電圧とする．そしてこれをブラウン管オシロスコープのY軸，X軸に引加する．そうすれば管面上に式(6.1)に基ずくHoughカーブが描かれる．この像を光学系を用いてＣＣＤイメージセンサ上に投影する．ＣＣＤイメージセンサの個々の感光素子は並列に積分コンデンサが接続されており、Houghカーブの通過度数がこのコンデンサに累積される．つまり物理的立体的にθ-ρヒストグラム平面のアドレッシングを行い度数累積の高速化を計ろうとするものである．これはまだ実験段階でハードウェア的にも大きな装置であるが将来的には有望な方法と考える．また筆者等の特許[1]は現時点では実用的な要求が少ないこととそのようなＩＣ回路の設計技術が確立されていないため製品として発表するには至っていないが本節の主目的を達成するための有効な手段である．これは電荷蓄積素子にハードウェア的に直接アドレッシングを行い，電荷を注入してアナログ的にヒストグラムを形成するものである．結果の読み出しはＣＣＤ転送ラインを経由して行う．

６．５　θ-ρヒストグラムの探索

θ－ρヒストグラムの形成が完了した後θ－ρヒストグラムの探索が行われる．この過程は非常にクリティカルな問題を含んでいる．というのは度数値の絶対値だけを頼りにしたのでは目的とする直線が短いとき累積度数が上がらず，検出不可能となる場合も少なくないからである．ヒストグラムをある度数の閾値で切ってそれ以上の度数の範囲だけを探索するような場合にこのようなことがおこる．そこで考えられるのがヒストグラムに２次元微分を施してその値が一旦正になった後，負に向かうときのゼロクロス点を度数ピークとするやり方である．これをθ，ρ両方向について行うのであるがかなり手順が複雑になる．本論文はソフトウェア演算で時間がかかる部分をハードウェアの並列処理性，高速性を利用して高速化を計ることを一つのテーマとしているがこのような手順はハードウェア化が困難である．むしろ先に述べた，閾値で切ってヒストグラム探索範囲を絞る方法がハードウェア化しやすい．第８，９章で述べる新しい手法ではこれを補助的に用いることを提案している．

目的とする直線の存在範囲がある程度わかっている場合には探索は非常に容易である．第１０，１１章の応用例はこれに属する．θ-ρヒストグラムの度数ピークが検出されたとき、その度数は直線の長さに比例する．この性質は実際の応用に当たって利用すべきパラメータの一つである．１１章の例ではこれを積極的に利用し欠陥抽出の手順の中に組み込んでいる．全く素性のわからない原画像からの直線抽出の際には結果を見て，原画像を一見しただけでは分からない直線の存在に気づくこともある．またこのHough変換で検出されるのは数学的に言う直線であって線分ではない．このこともHough変換を応用するに当たって十分注意すべき点である．

６．６　Hough変換演算の高速化

６．６．１　表参照方式と高速アクセス

図６．５はコンパイラ言語での配列宣言されたメモリの実アドレスとプログラムにおけるイメージの関係を示している．配列の次元はこの例では２次元である．第２添字は８とした．図の上方に実アドレス、下方に２次元配列としてのイメージが描かれている．

図からわかるように配列が char a[m][n];（Ｃ言語での表記）と宣言されると$m \times n$個の１次元の直列アドレス空間メモリが確保される．その時，配列の第２添字が先に変化する（図６．５左上参照）コンパイラが一般的である．その後ａ[i][j]が参照されると通常のコンパイラは$n \times i+j$の整数演算を実行して実アドレスを求める．

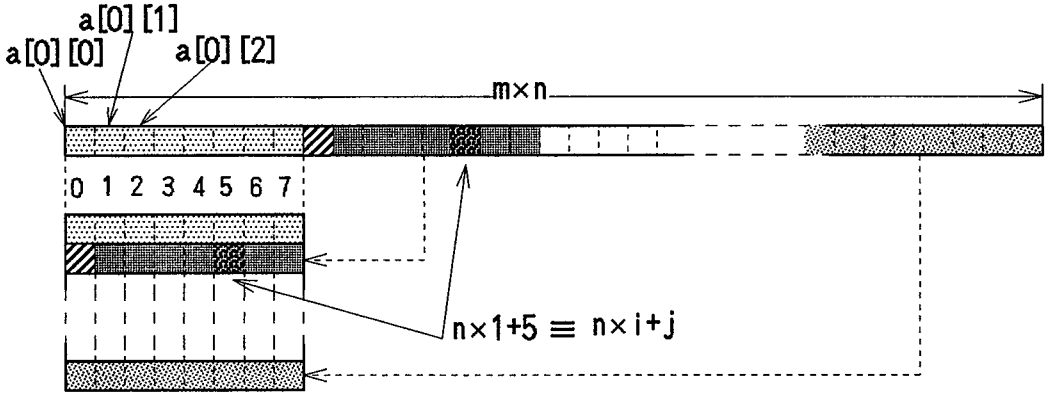


Fig.6.5 Physical image of array and software image

図では具体例としてi=1, j=5としてa[1][5]を参照している．このようにコンパイラレベルでは気がつかない整数乗算と加算が機械語レベルでは含まれているのである．これが３次元になると乗算２回加算２回の演算が必要となる．しかしＣ言語にはポインタと呼ばれる巧妙な手段がある．

```
char a[m][n];
char *pt;
pt=a[1];
```

と書けばptにはa[1][0]のアドレスが（図６．５の斜線部）代入される．従ってこの後シーケンシャルなデータの参照は *pt++; と記述しアドレスの単なるインクリメントで次のアドレスデータの参照が可能となる．インクリメント命令は通常の計算機では最も実行時間の短い命令の一つである．特にこのポインタをレジスタ変数に割り付ければ更に高速アクセスが可能となる．これを高速表参照の基本的な考え方である．詳細は次章で述べる．

６．６．２　分割縮小化による表サイズの低減

前項に示した表参照方式によるρの求値は高速ではあるけれども大きな容量の表を必要とした．この表サイズの問題をクリヤする今一つの方法として第７．３節で述べる分割縮小化表参照方式がある．ここではその概要を記す．

Hough変換の原式(6.1)における変数x, yのビット列を，次のように３ブロック（図６．６参照）に等分割して，それぞれ$x_0, x_1, x_2, y_0, y_1, y_2$と表現する．ここでは軸分割数は512（9[bit]）と仮定する．

$$x=(2^3)^2 x_2+2^3 x_1+x_0 \qquad ---(6.3)$$

$$y=(2^3)^2 y_2+2^3 y_1+y_0 \qquad ---(6.4)$$

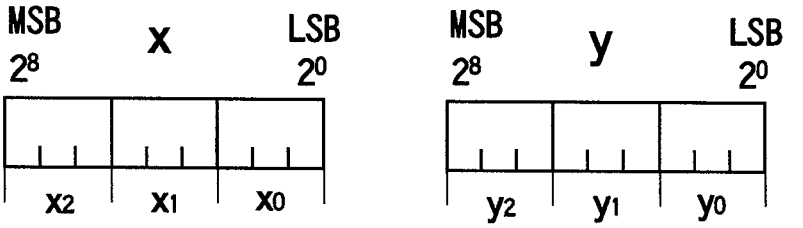


Fig.6.6 Dividing of variables

上式を式(6.1)に代入して

$$\rho = (2^3)^2(x_2\cos\theta + y_2\sin\theta) + 2^3(x_1\cos\theta + y_1\sin\theta) + (x_0\cos\theta + y_0\sin\theta) \qquad ---(6.5)$$

ここで

$$F(x_i, y_i, \theta) = \rho_i = x_i \cdot \cos\theta + y_i \cdot \sin\theta \qquad ---(6.6)$$

とすれば

$$\rho = (2^3)^2\rho_2+2^3\rho_1+\rho_0 = (2^3)^2F(x_2, y_2, \theta) + 2^3F(x_1, y_1, \theta) + F(x_0, y_0, \theta) \qquad ---(6.7)$$

となる．

従って$F(x_i, y_i, \theta)$の表を用意すれば，上式１，２項の乗算はそれぞれ２進数の６ビット，３ビットのシフト演算であるから，１つの表の３回の参照と２回のシフト加算によりρを求めることができる．x_i，y_i はそれぞれ３ビットであるから表の入力アドレス線はθの9本と併せて15本となり表サイズは2^{15}=32［k語］と縮小される．本章以降1［k語］は1024［語］，1［M語］は1024［k語］と表現する．

６．６．３　連立漸化式によるHough変換演算

ここではHough変換原式(6.1)を連立漸化式を用いて逐次的に高速計算する手順について述べる．この種の計算法は文献(23),(24)等において"FIHT"或いは"FIHT2"という呼称で初めて紹介された．そのρ値計算法は，穂坂衛著" コンピュータグラフィックス" [63]における円弧の簡易描画法に関するアルゴリズムに根拠をおいている．その手順の概要は次の通りである．

X-Y平面上のサンプリング点(x, y)が与えられたときρの初期値を$\rho_{1,0}=x$,　$\rho_{2,0}=y$,と

おきρ値を次の連立漸化式(6.8),(6.9) で順々に算出するものである．ρ_2は$n\cdot\Delta$が$\pi/2$だけずれた位置からのρの値である．

$$\rho_{1,n+1}=\rho_{1,n}+\Delta\cdot\rho_{2,n} \quad \text{---(6.8)}$$

$$\rho_{2,n+1}=\rho_{2,n}-\Delta\cdot\rho_{1,n+1} \quad \text{---(6.9)}$$

但し$\rho_{2,n}=\rho_{1,n+K1}$: $n=0,1,2,\ \cdots K_1-1$

ここでΔは角度の微少変化量で，$\Delta=1/2^m\fallingdotseq 0$　ここでmは正の整数である．

この方式はCPUのレジスタ間での演算でρ値を発生でき非常に高速である．しかし式(6.8)で求めたρが次の式(6.9)で使用され，直列演算の形式となっている．そこでこれを並列演算が可能なハードウェア上に組み込み，並列にρ値が算出出来るように変更し，その式の誤差等の検討を行っている．またこれを4重並列に計算する方法へも拡張した．これらの詳細な説明は第8章で行う．

6．7　結言

以上Hough変換の原理とこれを利用した図形形状パラメータ計測に関する問題を筆者が新しく考案した手法を紹介しながら述べた．以降の章ではこの章で概括した事柄の詳細を述べる．

7章　表参照方式によるHough変換

7．1緒言

Hough変換の開発の歴史，発展の過程については第1章において概要を記した．また第6章においてHough変換の画像入力から直線検出までの過程を分析し本章と第8，9章で述べる新しい方式の考え方を示した．その中で Hough変換を実際に応用する際，アルゴリズムの実行に要する時間が大なることが欠点であること，その実行時間を短縮するため表参照方式を採用すると時間は短縮されるが表サイズが非常に大きくなり，両者はトレードオフの関係にあること等を述べた．

本章ではまず 表を用いたHough変換演算の原理を示しその基本的な特長を述べ，その中で，どのような問題点が存在するかを明らかにする．そしてHough変換の過程の中でρ値を正弦関数の表を用いて高速に求める方法について考察し，2種類の方式を提案する．

下記にHough変換原式を再掲する．

$$\rho = x \cdot \cos\theta + y \cdot \sin\theta \qquad \text{---(7.1)}$$

ρはx, y, θなる3変数の関数であり，この3個の独立変数に，それぞれ配列の次元を与えて表にすれば$F[x][y][\theta]$なる3次元の表が必要になる．ここで鍵かっこで表される$F[x][y][\theta]$は表を意味するものとする．これはプログラミング言語" C "の記法にあわせたものである．まず従来の表参照方式を3種類ほど概観する．

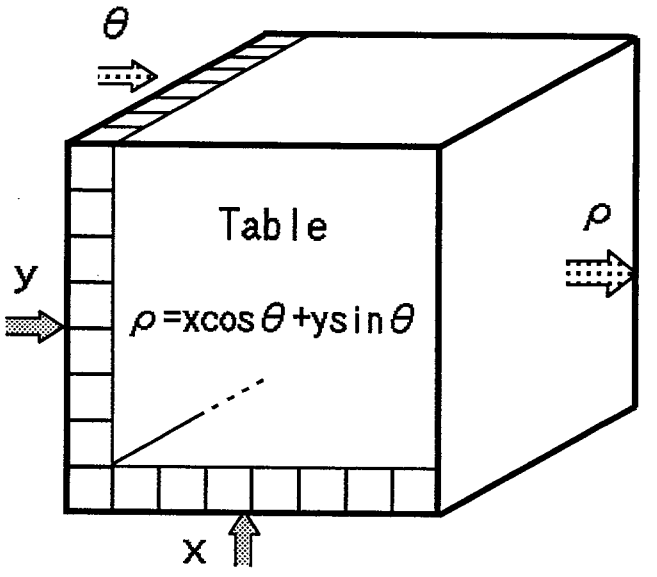


Fig. 7. 1 Three dimensional table

(a)完全表参照方式　（図7.1参照）

こらは上記$F[x][y][\theta]$なる3次元の表を用意する方法である．仮にx, y, θの軸分割数を512=2^9にとれば表サイズは$512^3=2^{27}\fallingdotseq$128[M語]の大きさを必要とし現実的でない．もし用意できたとしてこの表をランダムに索引しようとした場合，その実効アドレスの計算に時間がかかる．図7．2はこの様子を示している．つまり3次元の表の特定位置（$[x][y][\theta]$なるアドレス）を求めようとするとX,Y軸の寸法をそれぞれx_{max}, y_{max}とすればX-Y 2

次元平面のθ方向何番目の平面であるかを計算するのに($x_{max} \times y_{max}$)×θの計算が必要であり，その平面内のY方向行数を計算するのに($x_{max} \times y$)の乗算，さらにこの行内の列位置を特定するためにxの加算が必要である．($x_{max} \times y_{max}$)の値は予め計算しておけばよいから2回の整数乗算と2回の整数加算が必要なことが判る．このように表の索引には表てに出ない演算を内含している．これは後で述べるようにFORTRAN等の高水準言語で処理を記述する場合，実行速度の見積もりに大きな影響を与えるので注意を要する．

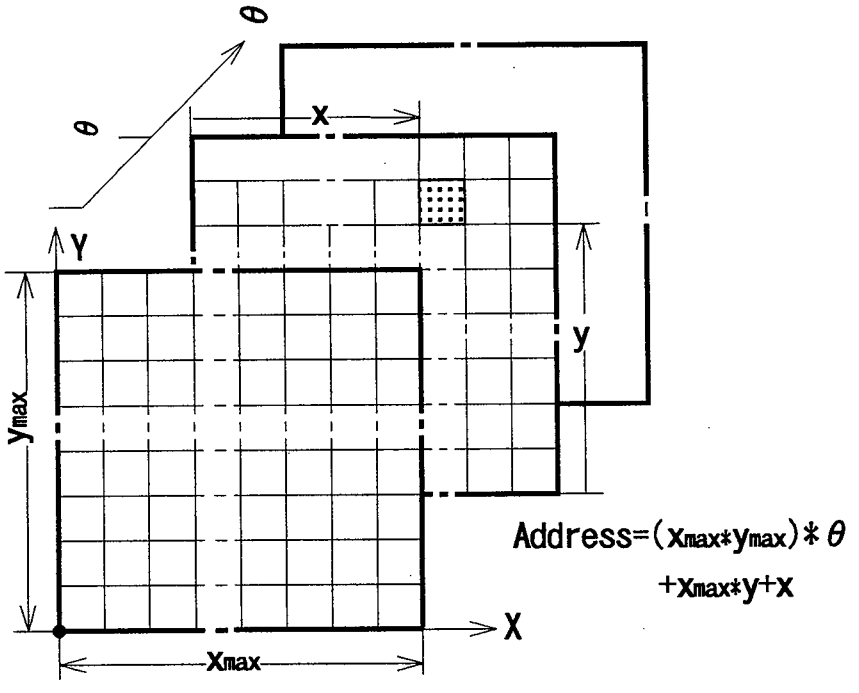


Fig.7.2 Calculation of effective address

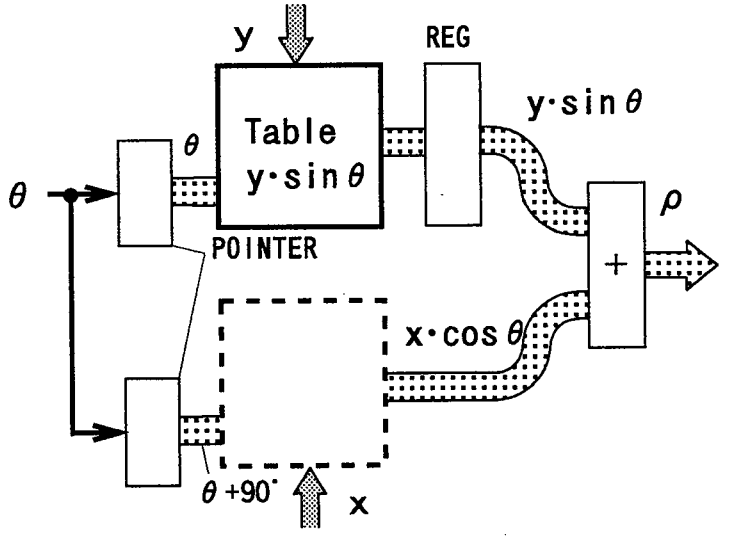


Fig.7.3 Two dimensional table

(b)２項分離方式　（図7.3)参照）

次に表の次元を１つ落として$F_x[x][\theta]$),$F_y[y][\theta]$なる２次元の表を用意することを考える．つまり　$x\cdot\cos\theta, y\cdot\sin\theta$を表として用意する．

$F_1[x][\theta]\equiv x\cdot\cos\theta$

$F_2[y][\theta]\equiv y\cdot\sin\theta$

として

$\rho = F_1[x][\theta]+F_2[y][\theta]$

より求める．この場合は $F[y][\theta]\equiv y\cdot\sin\theta$　を用意すれば

$\rho = F[x][\theta+90^\circ]+F[y][\theta]$　　---(7.2)

となり表は１個でよい．Fのみを用意すれば１つの表の２回の索引で演算可能である．

この結果としてF_1, F_2には次に１回の加算が追加されてρ値となる．ソフトウエア上で演算する限り，F_1, F_2の求値は直列に処理されるので表を１つにした方がよい．ハードウエア上でこれを行う場合，もし表が別々に用意出来るなら並列処理が可能であり，速度はこの部分では２倍になる．表サイズを概算すると，先と同じ条件で軸分割数512=2^9として256[k語]となる．これは現実に入手できるＩＣの容量である．しかしその後の加算処理が追加されることに注意しなければならない．

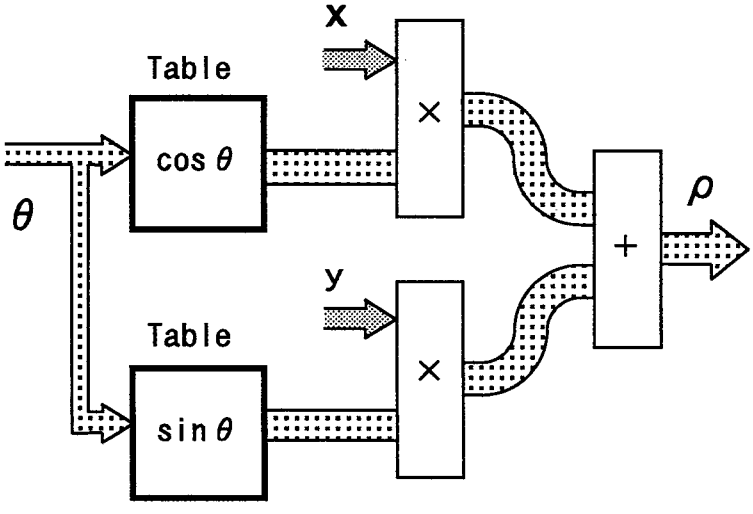


Fig.7.4 Trigonometric function table

(c)正弦関数表方式　（図7.4参照）

次に$F_1[\theta]=\cos\theta$と$F_2[\theta]=\sin\theta$の表を用意することを考える．90[°]ずらせば表は１つでよいことは前述と同じである．これ正弦関数表方式と呼ぶことにする．この場合，表サイズは極僅かで512[語]である．　しかしその後にxまたはyとの乗算が必要となる．乗算は速度低下の大きな要因となる．

これらの方式をHough変換を実用装置の中に組み込む場合，並列処理，パイプライン処理を行なう大規模な計算機資源を投入することは難しい．ミニコン，ワークステーション，高級パソコン程度が使われることが多い．このため処理速度の評価は，OSのオーバヘッドやマルチジョブで他の影響を受けない，スタンドアロンのシングルジョブシステムで行な

うのが望ましい.

θ-ρヒストグラムの形成手順は比較的単純な演算の繰り返しである．速度を上げるためにはループ内の命令の実行時間，バイト数（命令取り出し時間に関係する）等のきめ細かい検討が必要である．本章では通常の加算と，インクレメント(+1)を区別して扱う．通常の計算機ではレジスタのインクレメント操作は加算に比して高速だからである．その比は命令フェッチを含めて約 2:1 と思われる.

実時間システムの実用化を目指すには，必要な計算量からして，ハードウエア化が必須である．このためハードウエア化する際の問題を考慮して最適化を検討すべきである．例えば回路設計の容易さ，回路規模，並列処理への展開可能性等である.

ところで表を参照する際の実際問題として$F[x][y][\theta]$のθは+1しながら索引されることを考えると，表の要素の配置を工夫することによって先の整数乗算や加算を使わずに+1(インクリメント)だけで表を参照することが可能となることが推定される．この事は第6章で概説した．このような処理は言語”C”やアセンブラレベルで可能な方法である．言語”C”にはa++なる単項演算子が用意されておりコンパイラはこれを変数aのインクリメント命令に翻訳する．特にaをポインタとし，レジスタ変数に割り付ければ先の実アドレス計算のための乗算，加算が全く不要になり，且つワークメモリを参照する必要もなくなる．（このような記述はFORTRANでは指定不可能で，配列の添字に従って逐一，実アドレスを乗算，加算で計算してしまう．）この方法はソフトウエア，ハードウエア双方に適用可能であり，表としての参照メモリの縮小化も可能である．この種の方法を７．２節で考察する.

７．３節ではハードウエア上での処理に限定された，表の構成法”分割縮小化表参照方式”について述べる．これは表の独立変数となるx, yを表す２進数のビット列をいくつかに分割して表を用意する方式である．表参照の後，加算が必要になるがこれと処理速度が同程度の(b)２項分離方式の表より表サイズが小さくなる.

７．２　極座標型表参照方式高速Hough変換

７．２．１あらまし

表参照方式はρ値計算上，他の方法に比して最も高速であるが，表が占有するメモリ量が大きくなる．しかし，Hough変換を実時間システムに応用する際の最大のネックは速度の問題であり，メモリコストは下がる傾向にあるから最も有望視される．本節では７．１節で述べた表構成による表参照方式について考察した結果を記す.

表参照方式について演算ステップ数，演算時間，誤差等の面から検討を加え，Hough変換の基本式を変形し，r, ϕ（後述）なるパラメータを用いた表の構成が良いとの結論を得た．この構成はソフトウエア処理，ハードウエア処理のいずれにも適用可能である．ソフトウエア処理での効果を確認するため数値実験を行なった結果，基本式のままでのHough変換演算に比して時間は1/80であることを確認した．従来からある表参照方式は前節で説明した様に，大きく３つに分類されるが表容量と処理速度は相反する関係にある．本節の提案する方式では処理速度を同じにした場合，表容量が1/100以下に，また表容量を同一にとった場合速度は約 2 倍になる．表参照方式に関して，スーパーコンピュータを使

ったシミュレーション，文献(25)では配列のアドレス計算時間やOSのオーバーヘッドタイムによる高速化の頭打ちが指摘されている．本構成法によればオーバーヘッドタイムの影響は同じであるが，読みだすアドレスが連続しているのでポインタのインクレメント(+1)操作でメモリをアクセス可能で（言語”C”を仮定），配列のアドレス計算に要する整数乗算と加算が不要となり，2次元配列をランダムアクセスするのに比べ高速化できる．実験ではその比は約 2.3 倍であった．(32ビットパソコンでの結果---ポインタをレジスタ変数に割り付けた．)

本節では新しく提案する方式について詳細を説明し，その優位性を述べる．次にソフトウエアでの模擬実験結果について述べる．ついで試作したハードウエアの概要を述べ，回路上の工夫，その構成と機能評価結果を記す

図７．５に本文で用いるX-Y平面，θ-ρ ヒストグラム平面の座標構成を示す．図中J, K, L, Mは各軸の分割数である．

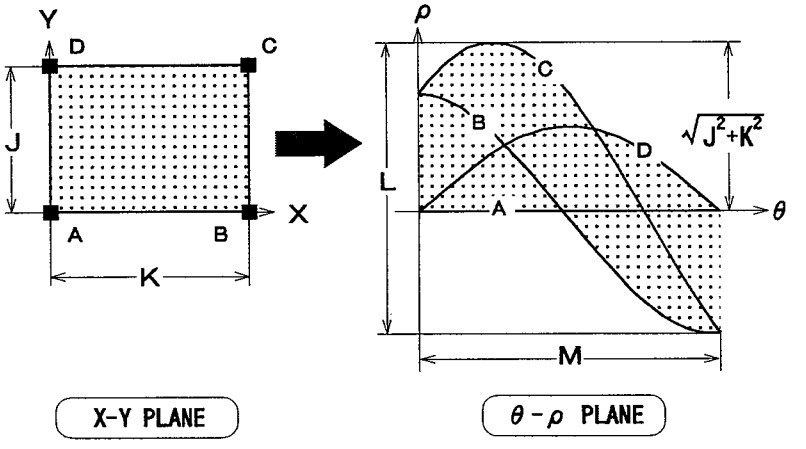


Fig.7.5 Dimension of X–Y plane and θ – ρ plane

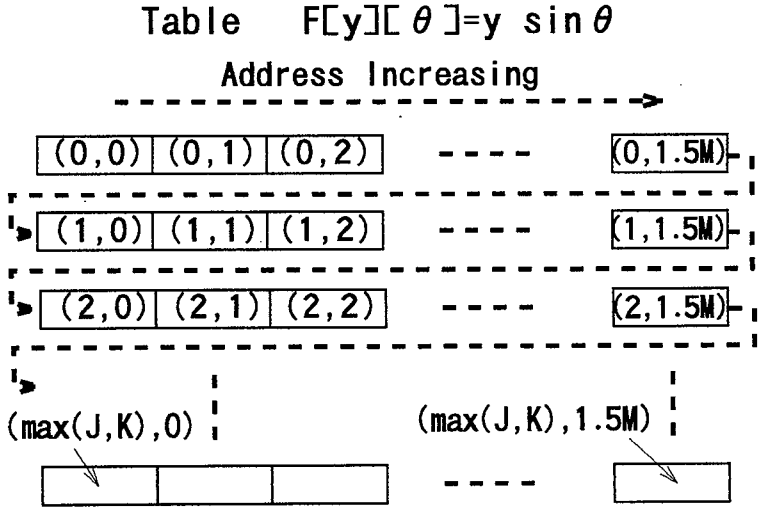


Fig.7.6 Memory address assignment of look–up table

分割数を全て同じにとるときはDで表す．ソフトウエアに言及するときは言語は”Ｃ”を用いる．

ここで２項分離方式についてさらに検討を行う．式(7.2)の$F[y][\theta]$はθの0～πの変化に対して0～3π/2に対応する必要があるから0～πに対して1.5倍となる．従って表容量はJ,Kの内で大きい方をmax(J,K)とすればmax(J,K)×M×1.5［語］，またはD^2×1.5［語］となる．ここで１語のビット長はθ-ρヒストグラム平面の１点の最大累積度数を決定する．θ-ρヒストグラムの形成中x,yは固定している．そこでアドレスを図７．６の如く割り振り，θ及び$\theta+\pi/2$をそれぞれポインタ変数θ_1，θ_2に割当て初期値をそれぞれ 0°，$\pi/2$とする．このようにすればθ_1，θ_2をインクレメントしながら１つの関数表を用いて目的の値を得ることができる．

この方法は次の点で問題がある．

①ループ中に式(7.1)の２項の加算が含まれる．（速度の問題）

②誤差の関係から整数加算は不利．

③ハードウエアに組み込むとき，表を２回読み出すのでアドレス線の切り替えが必要であり，速度の低下につながる．

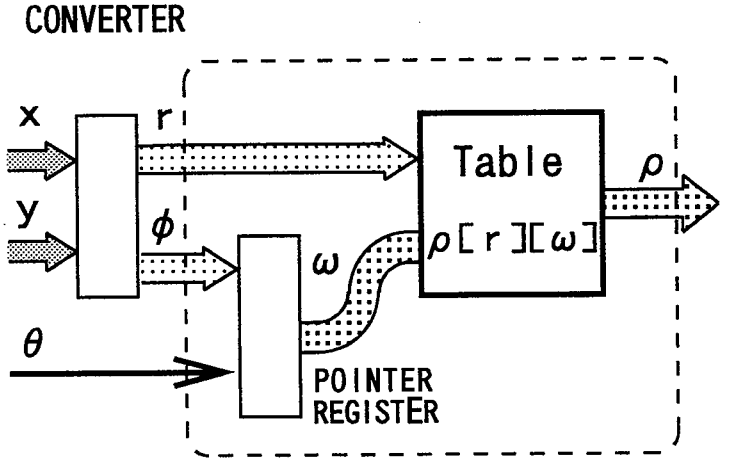


Fig.7.7 Proposed new table look-up method

７．２．２ 極座標型表参照方式の原理　　--図7.7参照

次に本文が提案する方式について手法の説明と有効性について述べる．表参照の基本的な考え方は前章６．６．１項で示したがここではハードウェア化とも関連させて詳細に説明を行う．まずHough変換の基本式(7.1)を変形基本式(7.3)のように変形する．

$$\rho = r \cdot sin(\theta + \phi) \qquad ---(7.3)$$

ここで

$$r = \sqrt{x^2+y^2} \qquad ---(7.4)$$

$$\phi = tan^{-1}(x/y) \qquad ---(7.5)$$

図７．８は変形基本式の意味付けを示す．rはHough変換点(x,y)から原点までの距離である．このようにρを長さ r，と角度($\theta+\phi$)で表すのでこの方式を**極座標型表参照方式**と呼んでいる．rとϕの計算はX-Y平面の変換点一個につきθの変化にかかわらず１回であ

り，計算量の増加は無視できる．専用ハードウエア化は図７．７の破線枠内だけに対して行う．このときはホストコンピュータでは変換点のr，ϕの計算を行い，これを送出後，すぐに次の点のr，ϕの計算を行う．このことにより専用ハードウエアでのθ-ρヒストグラムの形成との並列処理が可能となり効率が向上し，また構成もシンプルとなる．表は$F[r][\omega]\equiv r\cdot sin(\omega)$を用意する．

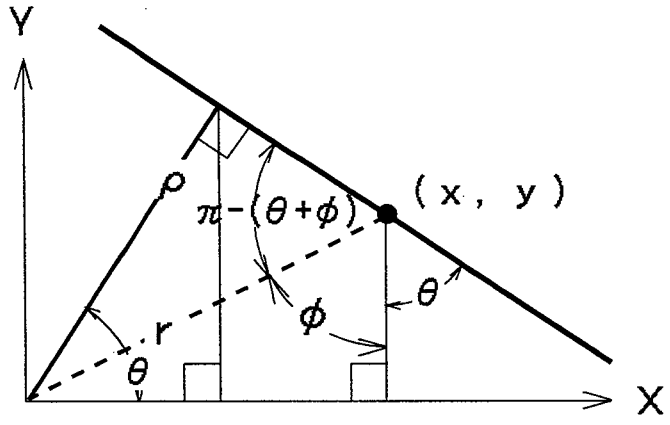


Fig.7.8 Relation between (θ , ρ) and (r, ϕ)

ただし$\omega=\theta+\phi$である．すなわちrとωをアドレスとしρの値を内容とする正弦関数の表を用意する．この表のアドレスの割り振りは図７．６と同じように行なう．図７．９はこの様子を示している．表は$\rho[r][\omega]$なる２次元の表を用意しその内容を太線の正弦曲線で示している．rが大きくなるに連れてその振幅が比例して増大している．rは表の参照中固定しているので，ρの値はθをポインタ変数に割当て，初期値をϕとしてインクレメントしながら表を連続的に読み出せば得られる．アドレス計算では加減乗除算が含まれず，+1 操作のみでよい．表に ROM(Read Only Memory)またはSRAM(Static Random Access Memory) を用いるときにはソフトウエア処理と同様にωを下位側アドレスに割り付ける．ϕの値が0～$\pi/2$の範囲で変化するので，方式(b) ２項分離方式と同様にωとしては0～$3\pi/2$に対応しなければならない．ρの計算に加減乗除算，関数計算が全くないので (b) の方式の加算時間の問題が解決されている．表サイズは直接的には$\sqrt{J^2+K^2}\times M\times 1.5$である．ここで$\sqrt{J^2+K^2}$は $r\cdot sin\omega$ に於けるrの分割数に対応し，Mは$sin\omega$の分割数に対応する．$\sqrt{J^2+K^2}\geq J$, K であるからこのままではρ軸分割数は増加する．しかし積の精度は低い方の項の精度に支配されることを考慮すると，$\sqrt{J^2+K^2}$個への分割は分割数の増加のみに留まり精度向上の効果は期待できない．そこでJまたはKの大きい方をとり，表容量を$\max(J,K)\times M\times 1.5$［語］に圧縮し，各軸分割数を全て等しく$D=J=K=L=M$とすれば，表容量は$D^2\times 1.5$［語］となる．もともとメモリは原画像用として$D^2$語，$\rho$，$\theta$平面用として$D^2$語用意する必要があるから，この程度の表容量は大きな負担ではない．D=512(9ビット)とし最大度数を8ビットで表すなら$512^2\times 1.5$=384［kbyte］の容量である．D=256(8ビット)なら96［kbyte］であり，これはパソコンでも十分実現可能なサイズである．

図７．１０は表の内容(a)と読み出される結果(b)の関係を示している．$r=r_1$のときには初期位相ϕ_1から$\phi_1+\pi$まで読み出され(b)のr_1曲線となる．同様にr_3に対してはϕ_3から

読みだされる．ハッチングの種類がそれぞれ対応している．

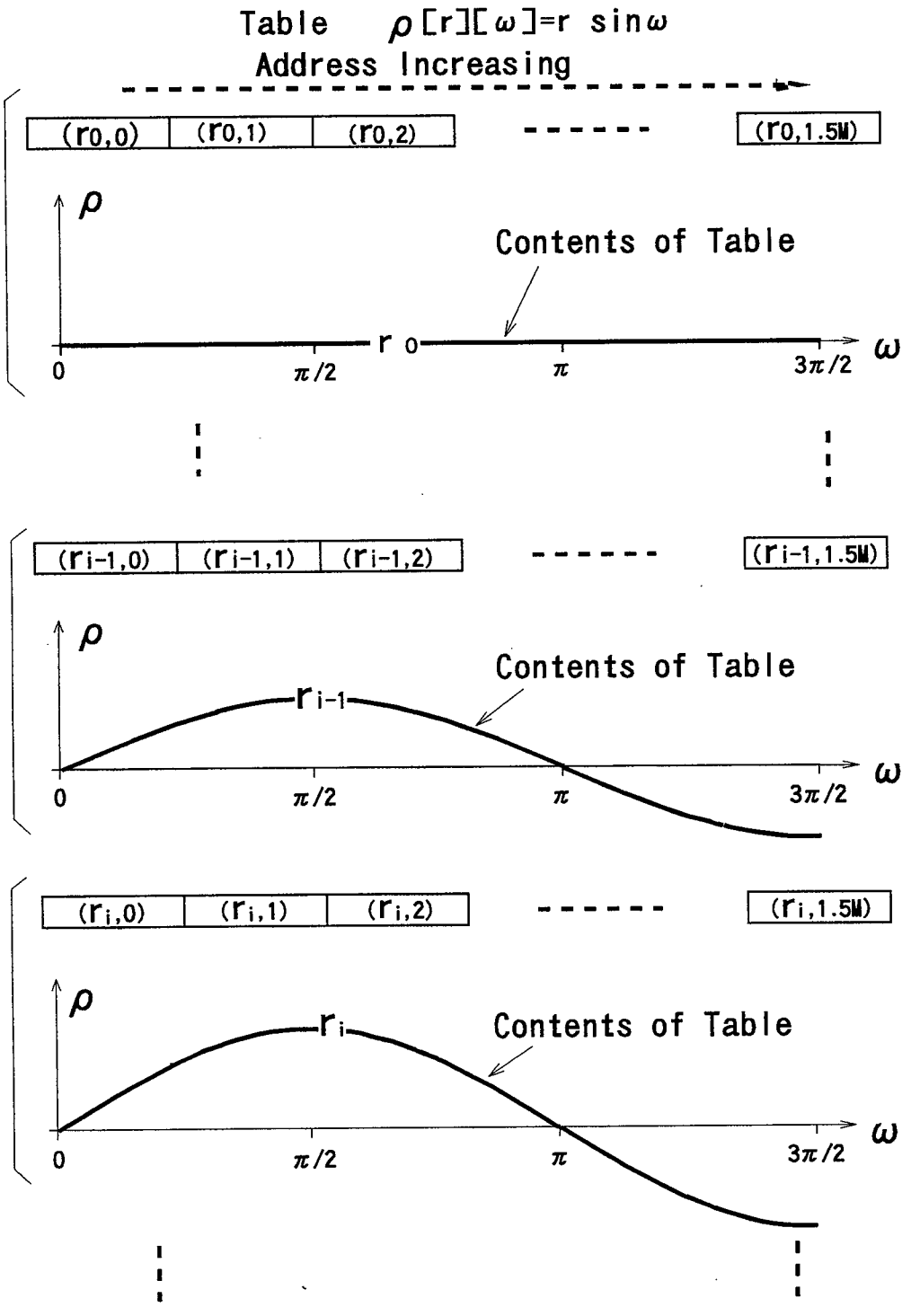


Fig.7.9 Table address and its contents

以上の検討に基づき本文が提案する，基本式を変形して表として利用する極座標型表参照方式を有効な方法と結論する．従来手法との比較においてその理由を要約すると次のようになる．本方式は

①７．１節(a)の方式に比し，必要メモリサイズがはるかに少ない．速度の点でも配列の実アドレス計算時間がないので速いと考えられる．

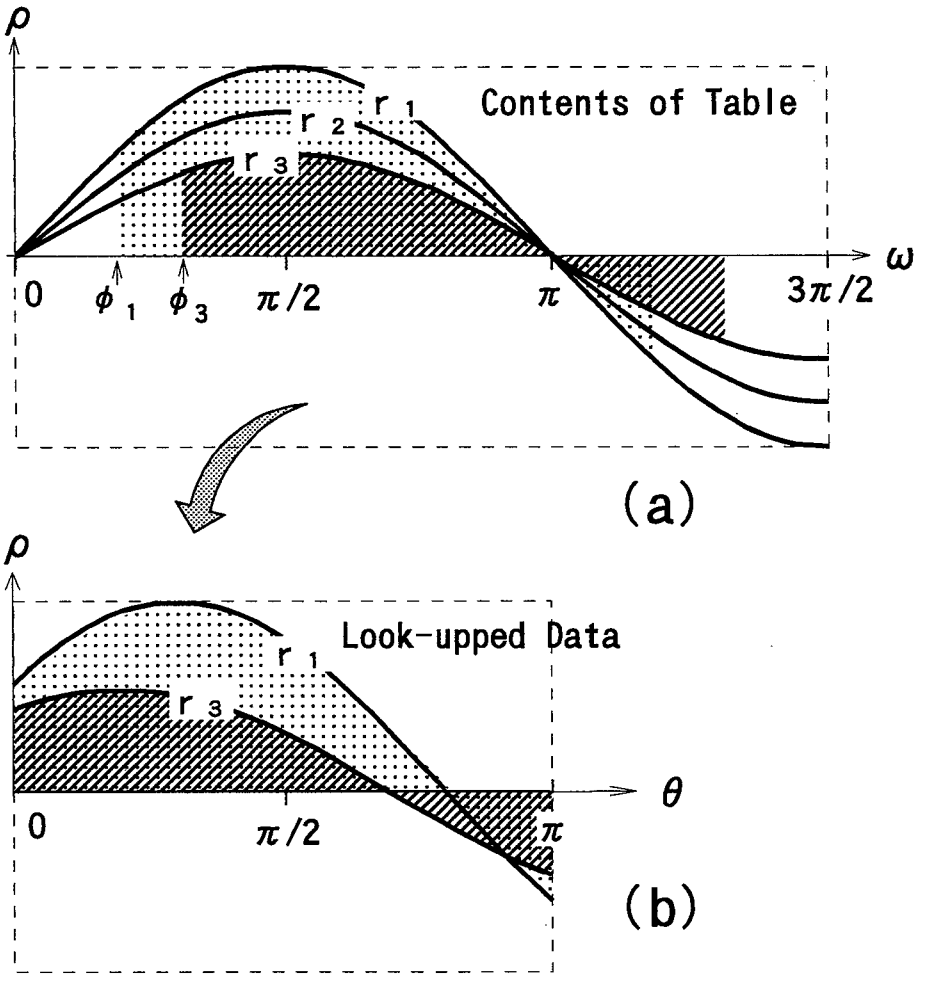


Fig.7.10 Contents of look-up table and look-upped data

```
void rtdraw(int x, int y)          /*x, y はサンプリング点座標  */
{char *st, *ste, *rtp, *rtb;
 unsigned int r, phai;
 r=sqrt(x*x+y*y)/1.41421356;       /* r を計算，基準化           */
 phai=atan((float)x/y)*M/3.1416;   /*φを計算，Mはθ軸分割数     */
 stb=&stb0[r][phai];               /*表読みだし開始番地を求む   */
 ste=stb+M;                        /*    〃    終了番地を求む   */
 rtb=&rtp1[0][0]-M;                /*ρ-θ平面先頭番地           */

 for(st=stb;st<ste;st++){          /*θのｌｏｏｐ               */
   (*((rtb+=M)+*st))++;            /*表読みだし，度数累積       */
 }
}
```

Fig.7.11 "C" function for θ – ρ plane drawing

②同節(b)の方式に比し，必要メモリサイズは同等であるが演算ステート数が少なく，結果的には速度が速い．

③同節(c)の方式に比し，必要メモリサイズは大きくなるが速度の点で問題にならない位速い．

そのほか全般的な問題として構成がシンプルでありハードウエア化も容易である．

7．2．3　ソフトウエアによる模擬実験

まず(7.1)式を直接計算する方法と，本文提案の方法について速度の比較をした．図7．11はHough変換点座標(x, y)を引数としてHough曲線を描く関数を”C”言語で記述した関数である．度数累積の核の部分をハッチングで示す．関数値索引とθ-ρヒストグラム平面における累積動作が1単項演算で行われ，加算が2回，+1が1回使われている．度数累積の正弦関数値の取り出しにポインタ変数を使用したように，θ-ρヒストグラム平面の形成にもこれを使用した．θ-ρヒストグラム平面を$H[\rho][\theta]$の形式で2次元配列としてアクセスすると実アドレス計算のための整数乗算と整数加算が1回ずつ入るからである．結果として式(7.1)の直接計算に比し約80倍の高速化が可能となり，θ-ρヒストグラム平面の1点あたり3μsで度数累積できた．次に，表容量がほぼ同等な 7．1節(b)の方式と比較した結果，1.6 倍の処理速度が得られた．(c)方式との比較は速度的に問題にならないと判断し省略した．実験に使用した環境はPC-9801FA (i486DX 66MHz), MS-DOS V3.3, Turbo C V2.1である．

7．2．4　ハードウエア構成

(1) 8ビット/語 ROMの9ビットシステムへの拡張について

軸分割数256(8 ビット)はクリティカルな値である．処理系の立場からは最も扱い易い値であるが処理される画像の精度からは 9ビットが望ましい．θ軸はメモリのアドレス軸なので番地数の大きいものを用いればよい．

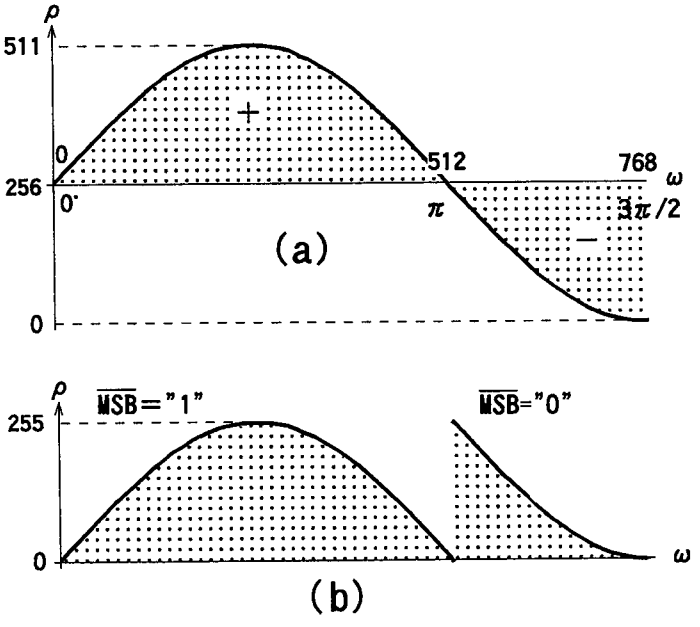


Fig.7.12 Contents of ROM

読みだされるデータであるρ軸に問題があり，ハードウエア化する上ではコストの面から8ビット/[語]のメモリを使用した方がよいが，次に述べる方法で軸分割数512(9ビット)のシステムを実現した．

図7.12(a)は普通に9ビットの精度で表を用意した場合である．ρ軸方向に 2^8 のバイアスを加えている．ωの値域 0° ～3π/2に対して 0～768 とすると，図よりω＜512まではρの値はρ≧256である．そこでROMの内容を図7.12(b)のようにωの値域で２分すると，8ビット/［語］のROMが使用できる．このとき ROM の出力は2^0～2^7の8本である．最上位の１本としてωの MSB を保持するカウンタを用い，これを反転してρの読みだし値の MSB とする．このようにすると 図7.12(a)と等価な表読み出しが可能となり，ω≦512の区間における 2^8 分の持ち上げが自動的に行なわれる．

（２） 回路構成

図７．１３に試作ハードウエアのブロック図を示す．r，ϕ，θカウンタは9ビットである．ωレジスタ/カウンタは10ビットである．度数積算用の加算器，レジスタは8ビットである．ROMは4［Mbit］(8［bit］/［語］)=512［kbyte］のものを１個用いた．実際データが書き込まれているのは3［Mbit］=384［kbyte］である．θ-ρヒストグラム平面用のSRAM(Static Random Access Memory)は1［Mbit］(8ビット/［語］)=128［kbyte］のものを２個用いた．度数は255までカウントできる．初期値0のセットと結果の読出しのためρ側のアドレスバスはホストコンピュータからのものと切り替えられる．ホストのメモリ空間に128［kbyte］単位のバンク切り替えで接続されている．θ側のアドレスはθカウンタで発生させる．ωレジスタ/カウンタの2^9ビットは反転されて，ROMと同じ量の遅れを持つ遅延回路を経てSRAMのρアドレス側2^8ビットに入力されている(前述の通り)．

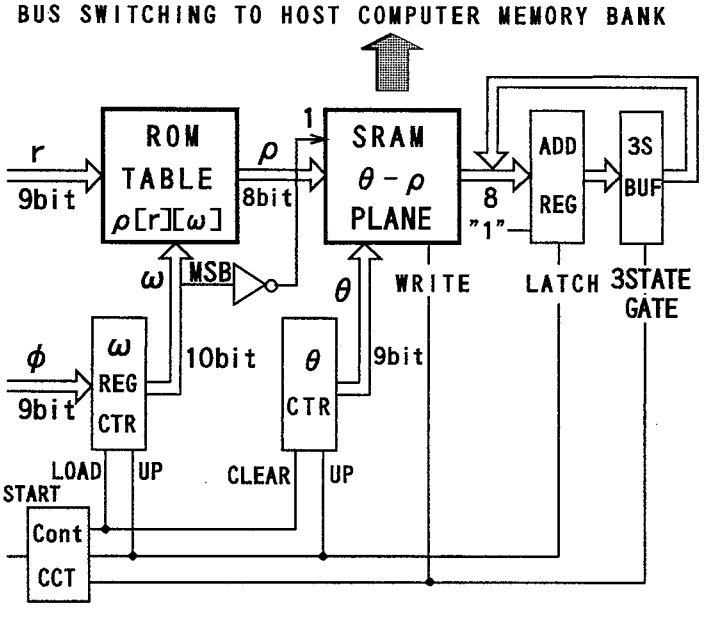


Fig.7.13 Block diagram of experimental hardware

論理素子はTTL 74ASシリーズ，ROMはサイクルタイム120ns，SRAMはサイクルタイム12nsのものを用いた．この構成でθ-ρヒストグラム平面(512×512画素)の１点あたり100nsで

ヒストグラムの形成が可能であった.

７．２．５　考察

（１）　処理速度の制限要因　－－（ハードウエア処理の場合）

試作ハードウエアのヒストグラムの形成ループ中のタイミングチャートを図７．１４に記す．表の読出し動作と +1操作は直列ではなく一部重複している．処理速度上のボトルネックはROM読出しに必要な120nsである．SRAMの内容 +1にはまだ余裕があり，実際に手にはいるICは保証規格値より性能的に多少優れていて，実験的には，タイミング調整によりθ-ρヒストグラム平面の１点あたり100nsで度数累積加算ができた(ρアドレス計算と度数書込みを含む).

この場合，最終的に速度を制限するのはROM出力データ遅れのばらつき(t_s)である．パイプライン処理ではROMのアクセス遅れ時間が出力データが"1"のときと"0"のとき，及び各データビット線で同じ値を持つことが保証されるなら(t_sが小さいなら)，遅れ量の絶対値(t_d)は問題にならない．このような要求を満たすROMの技術的検討が必要である．表としてSRAMを用いるなら（始動時に表データのロードが必要なことは言うまでもないが）速度を倍以上にすることは容易である．またDSPを用いて構成するよりも回路は簡素で且つ安価である.

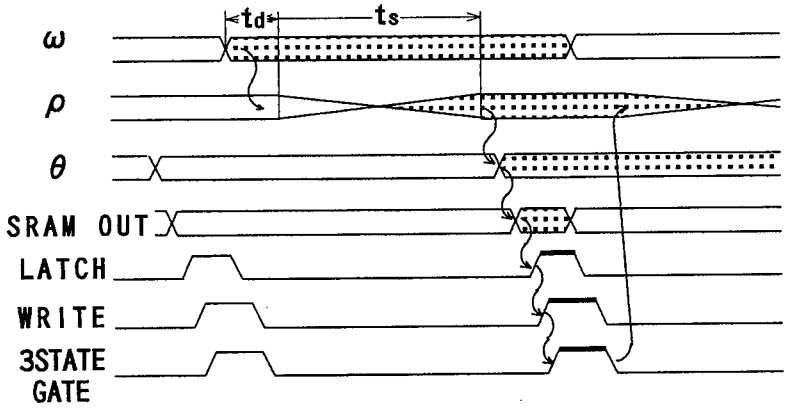


Fig.7.14 Timing chart for experimental hardware

（２）　表参照方式以外との比較

７．２節において本文提案の表参照方式が従来の表参照方式より優れているとの結論を導いた．ここでは １．２．３項で挙げた諸方式との比較を記す.

１．２．３項(a)はアナログ的に正弦関数を発生させる方式であり １％程度が精度の限界である．ビット長を増すことにより高精度化，高安定度化が可能なディジタル方式である本方式が有利である.

DSPを用いる方法と比較すると周辺回路を含めた全体回路量は増加し，方式的に単純な本方式が(b)より優れている.

(c)の方式は処理によって抽出される直線の位置などに制限を加えて高速化を図っている．本方式はこのような制約がない一般的な手法である.

文献(23)，(24)の中で沼田，興水等によるFIHT2[24]はレジスタ間の加減算，シフト演算のみで，メモリを参照することなく必要な正弦関数を発生でき，Hough変換をソフトウ

エア処理で実行するのに非常に有力な手法である[28]．しかし軸分割数に関係した一定量の誤差(軸分割数400の場合，最大誤差が1.5%---著者試算)が必然的に発生する．ここで言う誤差とはHough変換原式(7.1)との誤差である．一般的に，ソフトウエア処理とハードウエア処理の間には，処理速度に厳然たる差が存在するので，道路上センターライントラッキングによる自動車の自動運転のような実時間性が強く要求される応用では，処理のハードウエア化は必須であり，本法はこのような用途に適する．欠点は所要メモリ量が比較的大なることであるが，価格低下の一途をたどるメモリビット単価を考慮すると大きな問題ではないと思われる．

7．2．6　むすび

7．2節の内容を要約すると次のようである．

(1) Hough変換のθ-ρヒストグラムを正弦関数表を用いて高速に形成する際の表の構成法について検討した．その結果 Hough変換の基本式$\rho=x\cdot\cos\theta+y\cdot\sin\theta$を変形して，変形基本式$\rho=r\cdot\sin(\theta+\phi)$の形で表を用意する極座標型表参照方式を提案した．表サイズは原画像メモリサイズの1.5倍である．
(2) この方法と直接計算による方法との速度比較のためのソフトウエア実験を行った．パソコン上で処理時間の比は1:80であった．また同サイズの表を用いた従来の表参照方式との比較では 1:1.56 の速度比を得て本方式の優位性が示された．
(3) この方法に基づく専用ハードウエアを作成した．軸分割数を512とし，θ-ρヒストグラム平面1点あたり100nsで度数累積可能である．この分割数で表は市販のROM1個でよく，回路構成は非常に簡単になる．ハードウエア化に際して，8ビット／[語]のROMを用いて，9ビットに相当する軸分割数を実現する方法を工夫している．

提案の方法によってθ-ρヒストグラム平面形成の高速化が可能になった．今回，回路を手配線で組んだがＩＣ化されれば速度はさらに向上することになる．

7．3　分割縮小化表参照方式高速Hough変換

7．3．1　あらまし

本節では表参照方式で表容量の削減を目的とした新しい手法を提案している．これはハードウエア上でのみ実行可能な方式である．本方式によれば，小さい表容量で高速のHough変換が実現できる．また，そのハードウエア化も容易である．実験的検証のため作成したハードウエアでは，θ-ρヒストグラム平面の各軸分割数512×512，実効ビット数9ビットのとき，実効表容量は約74[kbyte]となり，同平面の1点当たり50nsの変換速度が得られることを確認している．

本節では，まず，提案する方式の原理を述べ，表容量の縮小化について論じている．さらにこの原理に基ずいた，ハードウエア化による演算速度の高速化について検討し，試作ハードウエアの回路構成と機能評価を示している．最後に，提案した原理について，やや一般的に考察している．

X-Y平面（原画像平面），θ-ρヒストグラム平面の座標構成とその分割数（座標点の数）は７．２．１項と同じで図７．５に示すとうりである．図中J, K, L, MはX, Y, θ，ρ各軸の分割数である同図(a)のX-Y平面内の各点A, B, C, Dは同図(b)θ-ρヒストグラム平面内の正弦曲線A, B, C, Dにそれぞれ対応している．

本節では，以下，軸分割数を$J=K=L=M=512=2^9$(9ビット)$=D$とし，すべて 512 に等しいとして議論を進める．従ってX-Y平面の各点(x, y)とθ座標の値域はすべて(0～511)として，9ビットで表すものとする．ρ座標の値はx, yと同じ単位で表すとすれば値域は($-512 \sim 511\sqrt{2}$)となり，整数部をそのまま表現すると，その語長は11ビットとなるが７．３．３項で説明するように，実際の装置では 9 ビットに制限している．

７．３．２　分割縮小化表参照方式の原理

（１）原理　原理の概要は第６章で述べた．結果として変数x, yのビット列(前述のように各9ビットとする．)を，次のように３ブロックに等分割して，（分割数が一般的な値の場合については後に論ずる．）それぞれ$x_0, x_1, x_2, y_0, y_1, y_2$と表現した場合，つまり

$$x=(2^3)^2 x_2+2^3 x_1+x_0 \qquad \text{---(7.6)}$$

$$y=(2^3)^2 y_2+2^3 y_1+y_0 \qquad \text{---(7.7)}$$

のとき

$$F(x_i, y_i, \theta)=\rho_i \qquad \text{---(7.8)}$$

$$=x_i \cdot \cos\theta + y_i \cdot \sin\theta \qquad \text{---(7.9)}$$

とすれば

$$\begin{aligned}\rho &= (2^3)^2 \rho_2+2^3 \rho_1+\rho_0 \\ &= (2^3)^2 F(x_2, y_2, \theta) \\ &\quad +2^3 \ F(x_1, y_1, \theta) \\ &\quad + \ F(x_0, y_0, \theta)\end{aligned} \qquad \text{---(7.10)}$$

となるので$F(x_i, y_i, \theta)$の表を用意すれば，上式１，２項の乗算はそれぞれ２進数の６ビット，３ビットのシフト演算であるから，１つの表の３回の参照と２回のシフト加算によりρを求めることができることがわかる．

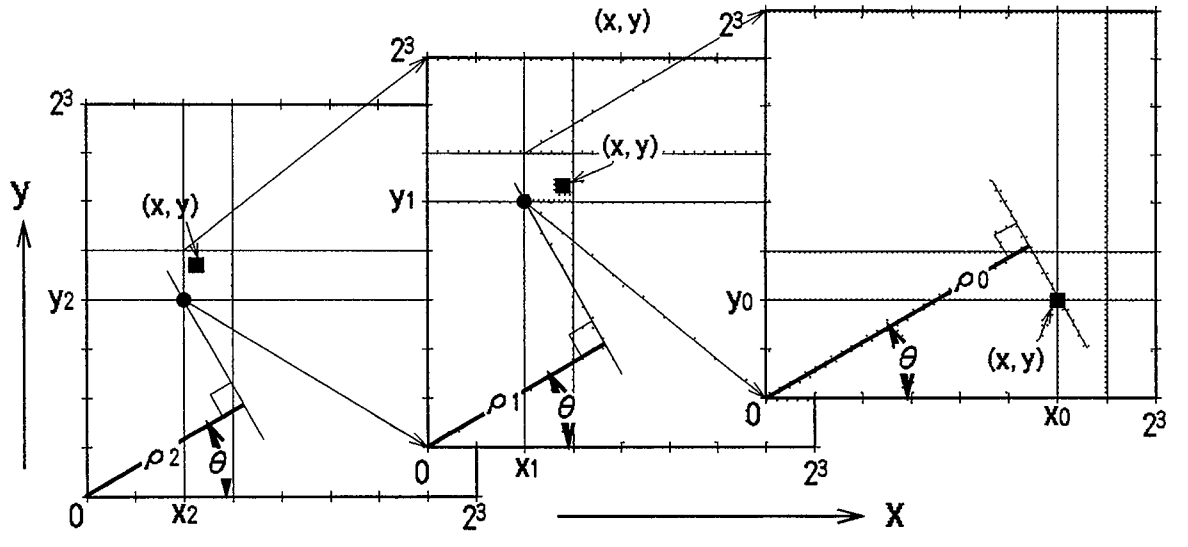


Fig.7.15 Meaning of variable dividing

（２） 変数ビットブロック分割の図解　　図7.15はビットブロックに分割化された変数x_i, y_i, ρ_i(i=0,1,2)の意味付けを図解して表している．図は変数x, y, θ, ρが式(7.1)を満足する値をとったときのもので，左から右に座標系は2^3=8倍ずつ拡大されて表示されている．図に見られるように，左図と中図（i=2,1にそれぞれ対応)においては，Hough変換はサンプリング点(x, y)が存在する小領域の左下部（原点）に対して行われることになる．

（３） ρの求値の数表化　　式(7.10)の $F(x_i, y_i, \theta)$ の表を用意すれば，式(7.10)１，２項の乗算は，それぞれ２進数の6ビット，3ビットのシフト演算であるから，１つの表の３回の参照と２回のシフト加算によりρを求めることができる． x_i, y_i (i=0～2)がそれぞれ３ビット，θが９ビットであるから，表容量は $2^3 \cdot 2^3 \cdot 2^9$=32［k語］で良くなり，７．１節(a),(b)よりははるかに小さくなる．　以上に述べた方式を変数ビット分割縮小化表参照方式，略して**分割縮小化表参照方式**と呼ぶことにする．

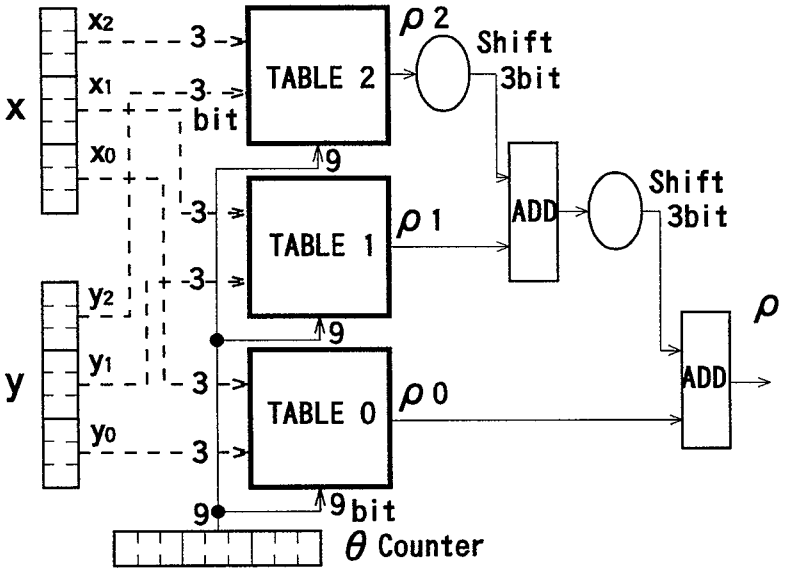


Fig.7.16 Principle of new table look-up method

（４） 表の複数化による処理の高速化　上記単一の表構成ではタイムシーケンシャルな３回の表参照と２回のシフト演算が必要になる．そこで，式(7.10)右辺3項それぞれに表を用意し，各表の出力をビットシフトを考慮して接続するような構成とすれば，3つの表の参照は並列操作で行われ，特別なシフト演算も不要になり，処理速度が向上する．図７．１６はこのような考えによる論理回路構成例である．表の数を３にしても表容量の合計は96［k語］（後述するように，実際の表容量はこの値よりは小さくなる．）と小さく，それ程のコスト上昇にはならない．図のように，この程度の容量の記憶素子に整数加算器２個を付加するのみの簡単なハードウエアで本文の原理は実現できる．

（５） 本方式の特長と他の表参照方式との比較　前述のように図７．１６において表2,1,0の語長は全て9ビット用意する必要はない．下位の表の下位ビットは最終出力としては切り捨てられるので，図の上方から下方へそれぞれ9,6,3ビットとなる．このようにビット長が表毎に変化するので厳密な単位としての[語]は決定できない．従って以後，容量を

示す単位[語]は概ね8ビット(1byte) として考える.

図７．１６は式(7.10)に直接対応させて描いたもので実際には表2,1,0の3出力は3入力の加算器により一挙に演算できる．この部分は組み合わせ論理回路であるからPGA(Programmable Gate Array)を用いて簡単に構成することが可能で，これによって処理速度は向上する．さらにこの部分はハードウエアで構成する際，表の読み出し，加算，θ-ρヒストグラム平面の形成とパイプライン処理可能である．速度の面では７．１節(a)完全表参照方式が最も速いが，本方式はこれと同等の処理速度を有し，且つメモリサイズを実用レベルに低減している．(b),(c)の方式は速度の点で本方式に劣る．

７．３．３　ハードウエア化の検討

シフトや加算の演算はソフトウエアでこれを実行すると，すべて時間的に直列の演算となるので，本方式は並行して処理が可能なハードウエア上で実行すると利点が大きくなる．そこでハードウエア上での実現を検討する．なお式(7.10)の$F(x_i, y_i, \theta)$をあらかじめホストの計算機上で計算し，その値を数表として当該ハードウエアのメモリに転送しておくものとする．

（１）X-Y平面及びθ-ρヒストグラム平面の座標設定　表にはROMまたはRAMを利用することを考えると，容量32［k語］は現在市販されているメモリICの中容量のもので十分であり，廉価に入手できる．そこで演算の並列実行のため，先に述べた如く表は式(7.10)右辺の3項それぞれについて用意する．表を個別に用意するものとすれば式(7.10)は次のように書き直される．

$$\rho = (2^3)^2 F_2(x_2, y_2, \theta) + 2^3 F_1(x_1, y_1, \theta) + F_0(x_0, y_0, \theta) \quad \text{---(7.11)}$$

ここで F_iの値が表出力ρ_i(i=0,1,2)に対応する．さて式(7.11)は式(7.1)の変形そのものであるから表の値は負の数になることもある．これを2の補数形式で表現するとF_1,F_0の語長が増大し，またρが負数になるとθ-ρヒストグラムの形成手順が煩雑になる．これを避けるために式(7.11)を以下のように変形し,座標系を変換する.

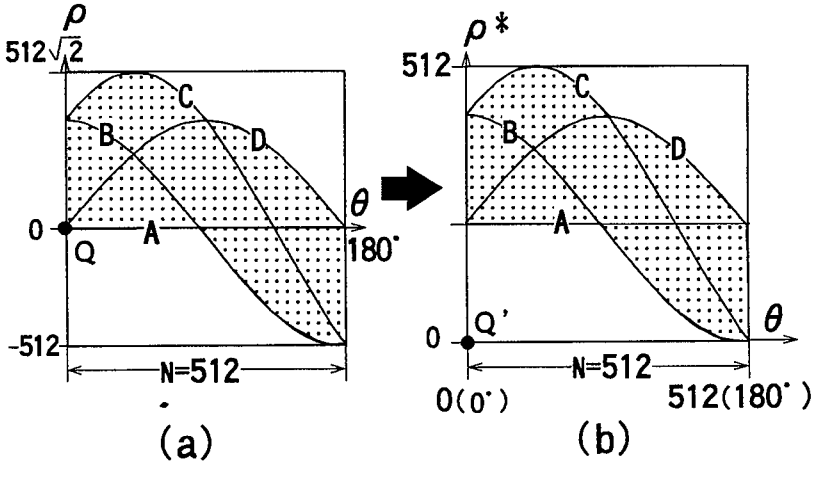


Fig.7.17 The coordinate and dimensions

そもそも，図7.5左図のような$X \geqq 0, Y \geqq 0$なる長方形領域内でρ値を計算すると，原点の

対角点Cのρの値が，ρの値域を支配することになる．今の場合，これは図7.17(a)に見られるように$-512 \leqq \rho < 512\sqrt{2}$である．そこで平行移動と圧縮をしたハードウエア系座標ρ^*を導入し，その値域を図7.17(b)のように$0 \leqq \rho^* < 512$(9ビット)に変換する．ρ^*は次式で与えられる．

$$\rho^* = (x \cdot \cos\theta + y \cdot \sin\theta)/(1+\sqrt{2}) + 512/(1+\sqrt{2})$$

これを変形して式(7.6),(7.7)を代入すると

$$\begin{aligned}\rho^* = \{ & (2^3)^2(x_2\cos\theta + y_2\sin\theta) \\ & + 2^3 (x_1\cos\theta + y_1\sin\theta) \\ & + (x_0\cos\theta + y_0\sin\theta) \\ & + 512\}/(1+\sqrt{2})\end{aligned}$$

結果として

$$\begin{aligned}\rho^* = & \underline{(2^3)^2(x_2\cos\theta + y_2\sin\theta + a)/(1+\sqrt{2})} \\ & + \underline{2^3 (x_1\cos\theta + y_1\sin\theta + a)/(1+\sqrt{2})} \\ & + \underline{(x_0\cos\theta + y_0\sin\theta + a)/(1+\sqrt{2})} \qquad \text{---(7.12)}\end{aligned}$$

但し　$a = 512/\{(2^3)^2+2^3+1\} = 7.01369$

従って

$$\rho_i^* = F_i^*(x_i, y_i, \theta) = (x_i\cos\theta + y_i\sin\theta + a)/(1+\sqrt{2}) \qquad \text{---(7.13)}$$
$$(i = 0, 1, 2)$$

以上の座標変換によりρ_2^*, ρ_1^*, ρ_0^*の語長はそれぞれ 9,6,3ビットに縮小される．合計表サイズは32［k語］の3倍にはならず実効的には(9+6+3)×32kビット，約72[kbyte]相当となる．シフト加算後ρ^*が整数になるためにはρ_2^*は固定小数点型の数値で整数部 3ビット，小数部6ビットの精度が必要である．同様にρ_1^*は整数部3ビット，小数部3ビットの精度でなければならない．ρ_0^*は整数部3ビットである．しかし，内容的には8のべき乗を乗じた値をそれぞれ用意すれば3個の表の出力を単に加算するだけでよく，基本原理に基づくシフト操作は不要となる．実際には式(7.12)の3つの項（下線で示す）の値を浮動小数点演算で計算し，最終的に四捨五入して表として書き込めばよい．

（２）　ρの値域を9ビットに制限する妥当性　　冒頭に述べた如く画像平面における軸分割数は512(9ビット)を基本としている．これは後で述べるように変数ビット分割数の関係で効率が良いこと及び，これ以下のビット数では実用的ではないと考えたからである．一方θ-ρヒストグラム平面のρの値域は前述のようにx,yの値域の$1+\sqrt{2}$倍に広がりその語長は11ビットとなる．ここでρ値の精度について考察してみる．式(7.1)のx,yの値を9ビットで表現すれば対象画像を前処理する過程で1/2画素分の相対位置誤差ε_1が生じるとして　$\varepsilon_1 = \pm 0.5/2^9$ となる．　$\cos\theta$，$\sin\theta$の値はホスト計算機中では浮動小数点数として取り扱われるのでε_1の精度よりはるかに高い．従ってρの相対精度は$2\varepsilon_1$に等しい．結果的にθ-ρヒストグラム平面内での累積度数ピーク点から求まるX-Y平面内の直線位置相対精度も $2\varepsilon_1 = \pm 1.0/2^9$ である．一方ρの値域を$1/(1+\sqrt{2})$に圧縮したρ^*では　x,y は$\pm 0.5/2^9$の相対精度を持ち式(7.12)の$1+\sqrt{2}$と　$\cos\theta$, $\sin\theta$の乗除算は浮動小数点演算で行われ，結果が整数に丸められて表(メモリ)に書き込まれることになるので，その精度ε_2

はほぼ $2\varepsilon_1$に等しくなる．θ-ρ^*ヒストグラム平面の累積度数ピーク点の値は$(1+\sqrt{2})$倍されてもとのρに変換されるが，この計算も前記の場合と同じくホスト計算機内における浮動小数点演算なので精度の劣化はない．従って値域圧縮による影響はほとんどないと考えてよい．このことはもともと512種類(9ビット)の変数値を，2048種類(11ビット)の値を取り得る空間に写像しても，取得る値がとびとびになり，2048種類用意する意義が失われると考えれば直感的に理解しやすい．現実にはパリティ用として1ビット追加した9ビット/語のメモリが市販されており，これを利用できる点も有利である．以上の理由により，ρの値域を9ビットに圧縮，制限することは妥当である．

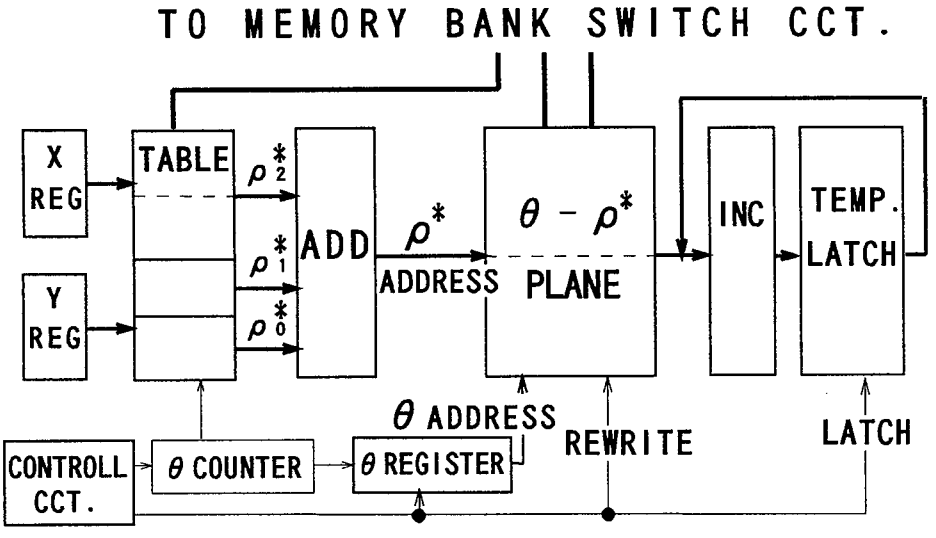


Fig.7.18 Structure of experimental hardware

7．3．4　ハードウエア構成

本方式の実験的検証のためハードウエアを作成した．図7.18はそのブロック図である．7．3．3項(1)で述べた9ビットのメモリはハードウエア試作時に入手困難であったので利用しないことにした．代わりに32［k語］，語長8ビット，アクセス速度17nsのSRAMをρ_0^*，ρ_1^*用として各1個，ρ_2^*用として2個の合計4個を用いた．このためρ_2^*用のSRAMの一方は1ビットしか利用していない．ROMを使わなかった理由はSRAMに比してアクセス速度が遅いからである．表データメモリとθ-ρ^*ヒストグラム平面メモリはホスト計算機のメモリアドレス空間の一部(128［kbyte］単位)と同一のアドレスに割り付け，実アドレス空間を別に用意したアクセスコントロールレジスタで切り替えて接続する，いわゆるバンク切り替え方式[16]を用いている．θ-ρ^*ヒストグラム平面メモリとしては128［k語］，語長8ビット，アクセス速度12nsのSRAMを2個用いた．同じくバンク切替えでホスト計算機のメモリと接続されている．θレジスタはθカウンタの出力を保持し，加算器からの出力遅れと同期をとり，全体の動作をパイプライン化するために設けている．

全体の処理の手順としては次のようになる．

(1)バンク切替えで表をホスト計算機のメモリに割当て，表データを書き込む．この後切離す．

(2)バンク切替えでθ-ρ^*ヒストグラム平面メモリをホスト計算機のメモリに割当て，内容を0にクリアする．この後切離す．
(3)サンプリング点のx, y座標を送出した後，対応するθ-ρ^*ヒストグラム形成を開始させる．その後次のサンプリング点を送出する．これを繰り返す．
(4)バンク切替えでθ-ρ^*ヒストグラム平面メモリをホスト計算機のメモリに割当て，θ-ρヒストグラムの探索を行なう．

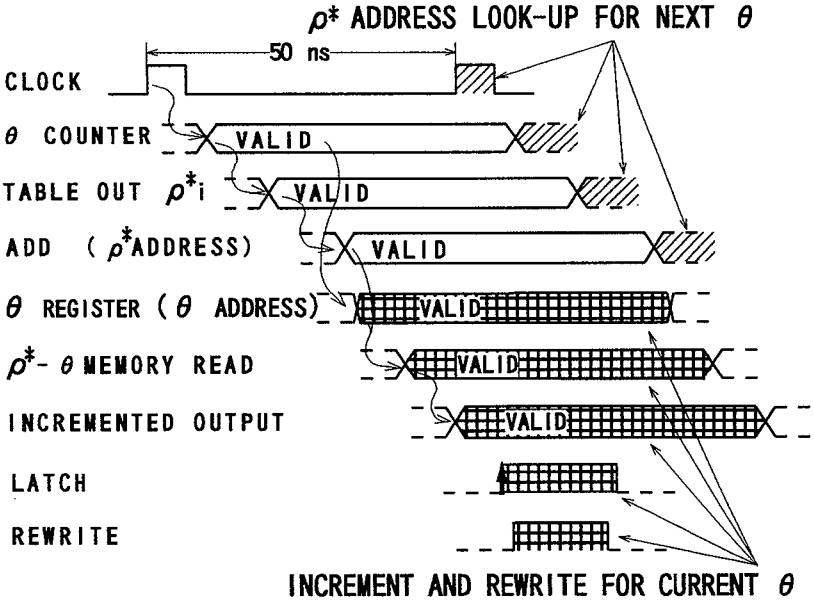


Fig.7.19 Timing chart

7．3．5　試作ハードウエアの評価

試作回路は理論通り動作し，方式的に問題がないことが確認できた．回路のICは高速部はTTL ASシリーズを用い，その他はALSも併用した．θ-ρ^*ヒストグラム平面の1点あたり処理速度Sは50nsと高速である．これは完全表参照方式と全く同等の速度である．図7.19は表読み出し，加算，θ-ρ^*ヒストグラム平面への書込みのタイミングを示している．図で濃いハッチング部は現θ-ρ^*ヒストグラムの形成処理を，斜線のハッチング部は次のθに対するρ^*の値の読み出しの処理を示している．このように表の読出しと，θ-ρ^*ヒストグラムの形成の手順はパイプライン化され，同時進行的に実行されていく．表としてROMではなくSRAMを用いたので参照動作が高速になり，両者の処理時間はほぼ同じでバランスしている．図7.20及び図7.21は本ハードウエアによる処理結果を示している．図7.20は変換前の原画像で，X軸に対して斜め 27°に置かれた長方形を計算機内で作成した．変換対象点の数は270個である．図7.21は変換後のθ-ρ^*ヒストグラム平面の度数分布で，図7.20のA，B，C，D辺が図7.21のA，B，C，D点と対応している．変換に要した時間は約14μsであった．

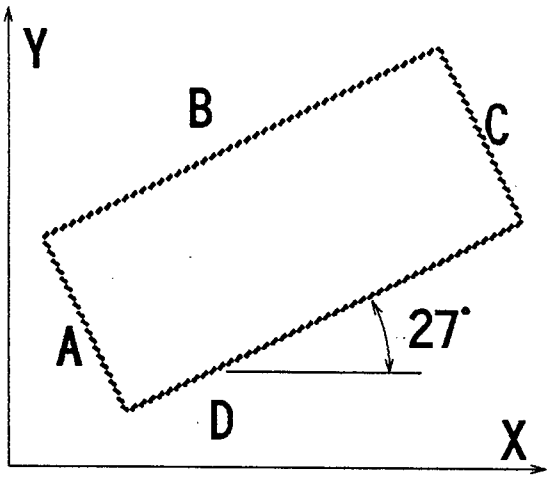


Fig.7.20 Given original image by computer for evaluation of hardware system

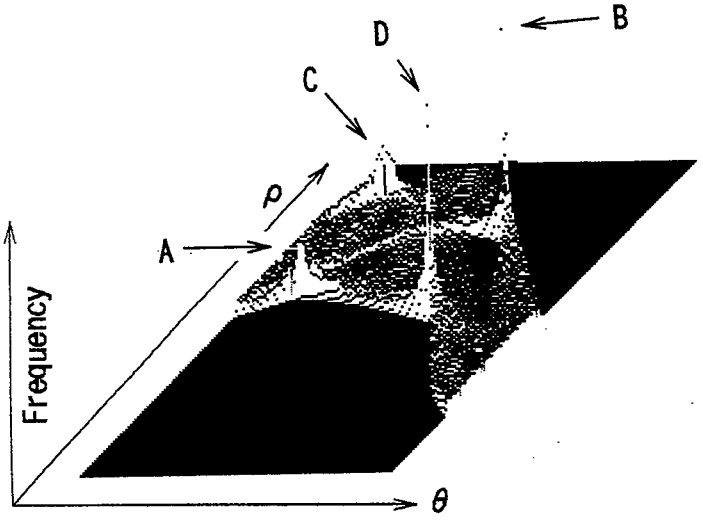


Fig.7.21 Result of evaluation test for hardware system
– Frequency distribution in $\theta - \rho^*$ plane after Hough transform –

7．3．6 考察

（1） 本方式で必要とする表容量の概略値　軸分割数をDとすれば図7.1における各表入力線の内，θ軸に相当する分は$log_2 D$本である．X及びYの入力線数も同じくそれぞれ$log_2 D$本である．表のX,Y側入力線数は変数ビット並びをそれぞれn個に分割すれば$log_2 D/n$の2倍となる．このような表がn個必要なので合計の表容量の概略値W[語]は次のようになる．

$$W = 2^{(2\log D)/n + \log D} \times n = D^{(2/n)} \times D \times n \qquad \text{---(7.14)}$$

先に述べた如く，低位の表ほど精度を落とすことができるのでn=3の場合，平均すれば(9+6+3)/3=6 ビットであり，全体の平均語長を8ビットと仮定すれば6／8に減少するので

$$W \fallingdotseq D^{(2/n)} \times D \times n \times 6/8 \fallingdotseq 72[\text{kbyte}] \qquad \text{---(7.15)}$$

となる．分割数 n を増せば W は急速に減少するが，n 入力の加算器が必要になる．n は D の値との整数徐算の関係で($log_2 D$)/n の剰余が0のときが最も効率が良い．

（２）分割数ｎの最適値　式（7.14），（7.15）より変数のビット分割数nを増せば表サイズは減少するが，加算回数が増加し速度の低下につながる．極端な場合としてn が変数のビット数に等しいとき，加算器群は乗算器そのものとなり，７．１節(c)に述べた方法に等価になる．したがって,n の最適値が存在するはずで，これを決定する条件は次の通りである．

①n の値の上限は加算回路の複雑さ，加算に要する時間で抑えられる．

②n の値の下限は入手可能なメモリICの容量，価格，納期で抑えられる．

③変数のビット数をn で除した剰余が0になる方がメモリのアドレス空間としての使用効率がよい．また市販されているメモリICの語長との関係を考慮する．

たとえば軸分割数256（変数ビット数8）のシステムではn =2として，アドレス線数16本のメモリ(64［k語］)をρ_0*, ρ_1*として2個用いた方が良い．ただし語長は8ビットと4ビットである．(語長4ビットのICは規格品がある．)

（３）メモリ素子の語長の使用効率　今回の試作ハードウエアでは表として32［K語］(語長8ビット)のメモリIC4個を用いた．その内訳はρ_0*と，ρ_1*としてそれぞれ1個，ρ_2*として2個用いている．その中でρ_0*は8-3=5ビット，ρ_1*は8-6=2ビット，ρ_2*の内の1個は７．３．４項で述べたように8-1=7ビットが使用されていない．効率は良くないが実用システムとしては512の軸分割数が必要であることと，メモリICを同一種のもので統一した方が回路の簡素化が図れることがその理由である．

（４）他の専用ハードウエアとの速度比較　本研究以外に花原ら[22]，恩田ら[34]，Rhodesらによって専用ハードウエアによるHough変換が試みられている．文献(28)によればθ-ρヒストグラム平面の１点あたり処理速度S はそれぞれ500ns，62.5ns，50nsとなっている．本研究での成果はS =50nsが得られたが,恩田，Rhodesらのシステムがメモリをそれぞれ8分割,4分割して，並列処理を行っていることを考慮すると方式的には本法がより高速であると言える．本法にこの種の並列化手法を取り入れることは容易であるから並列度に応じたより一層の処理速度の向上が期待できる．

（５）高速インクレメンタルHough変換法FIHT2との比較　表や関数計算の専用ハードウエアを用いずにρの値を発生させる方法としてFIHT2[24]がある．この方法は２元連立漸化式により，ソフトウエアで逐次的にρを計算する優れた方法であるが，前式の結果が次の式で用いられ，そのままではハードウエアによる並列処理に適していない．したがってソフトウエア処理だから，速度的には限界がある．また漸化式であるため，x, y, θの値が大きい位置で若干誤差が増大する．FIHT2をハードウエア化して処理速度向上を計る場合にはこれらの問題が解決されねばならない．この件については第８章で詳述する．

（６）７．２節の表を２次元化する方法との比較　７．２節及びV.F.Leaversの報告[49]では基本変換式(7.1)を

$$\rho = \sqrt{x^2+y^2}\ cos(\theta - tan^{-1} y/x) = r \cdot cos(\theta - \phi) \qquad ---(7.16)$$

と変形しρをrとϕの２変数の関数にすることにより表サイズを縮小した．この方式は試算すれば，軸分割数512の場合約384［kbyte］となり，かなりの縮小はされるが，本節で

提案する方式より数倍の大きさの表が必要である．またrとϕの値を計算する必要が生じ，ハードウエア構成が複雑になる．速度の点では同程度であると考えられる．ただし本節の方式はハードウエア上でのみ実行可能な点が異なる．

７．３．７　まとめ

７．３節の内容を要約すると次のようになる．

(1) 最も高速な３次元完全表参照方式を改良し，画像平面内の変数の２進数表現ビット列をいくつかのブロックに等分割した表を用いることにより，表サイズを実用レベルに縮小して実現する”分割縮小化表参照方式”を提案した．

(2) 提案の方法を実際の試作ハードウエアに実現して，方式の正当性を検証するとともに，処理速度の評価を行った．結果として，完全表参照方式と全く同等の速度が実現できた．分割のトレードオフとして加算処理が追加されたが，θ-ρヒストグラム平面メモリの加算処理をパイプライン化することにより，速度の低下は認められなかった．

(3) 表サイズは３次元完全表参照方式の表と比較して３桁程度小さく，市販の中容量のメモリが使用可能である．

(4) 試作ハードウエア上で処理速度はθ-ρヒストグラム平面の１点当たり50nsであった．

本文提案の方式を採用することによりHough変換の実時間システムへの応用が可能となるであろう．従来からある，図形の性質を考慮して変換対象点の数を限定する方法（１．２．３項(c)）等と組み合わせれば，相乗効果により更なる高速化が期待できる．今後は画像入力からエッジ抽出，θ-ρヒストグラム平面の形成，同平面の探索によるピーク点検出等を含むトータルシステムとしての検討が行なわなければならない．

７．４　結言

本章ではHough変換のρ値計算過程で正弦関数計算による速度の低下を減少させるため，予め関数値を計算記憶した表を用意しておき，この表を参照することにより，計算時間の短縮を図る２種類の方式を提案した．その解析の過程で表面的には最も高速であるはずの完全表参照方式（７．１(a)）で想定したほどの速度向上の成果が得られないことを示した．

ここで本章で提案した極座標型表参照方式と分割縮小化表参照方式を比較してみる．両者とも専用ハードウェアを試作したが極座標型表参照方式がメモリ容量384[kbyte]，サンプリング点一点当たりの処理時間は100[ns]であった．分割縮小化表参照方式はメモリ容量72[kbyte]処理時間は50[ns]であった．前者の処理時間100[ns]は表として使用したＲＯＭのアクセス速度による制限であってこれは後者と同様にＳＲＡＭを用いれば処理時間50[ns]は達成可能である．従って処理速度はθ-ρヒストグラムの形成に要する時間で決定される(７．３．５項参照)．一方ハードウェア構成上は前者がメモリを約５倍多く必要とする．しかし後者は３入力の加算器を付加しなければならない．従って一律にどちらの方式が優位であるかは断定できない．実際の実用装置設計の段階でメモリが安価か、加算器を付加した方が安価であるか，或いは納期その他の諸問題から方式を選択する必要がある．

本文の中には示さなかったが最近では浮動小数点乗除算の高速化のために専用ハードウ

エアが備えられており、一部の計算機では整数の乗除算より浮動小数点演算の方が速いというものもある．トータルなHough変換計算演算速度についてもこのような一部の計算機では７．１節(b)或いは(c)の方式が同節(a)の方法より速いという結果が得られたものもあった．しかしこれは極一部のマシンでの結果であり一般的なものではないとして本文では言及していない．本文では２種の表参照方式をとりあげたが他の高速Hough変換法と比較し速度的には優位にあることは疑う余地がない．現在この方式の処理速度を制限している要因は表としてのＲＯＭ(Read Only Memory)のアクセス時間である．最近のパソコン、ワークステーションのＣＰＵの速度向上には目を見張るものがあり、それに対応してメインメモリとしてのＤＲＡＭ（Dynamic Random Access Memory）の速度も速度の向上がなされている．これに対してＲＯＭの速度向上には関心が薄いように思われる．これは一つは需要の少なさとの関係からであって、いずれ市場に普通に出回っているＳＲＡＭ程度(8ns前後）には速度向上がなされるであろう．また現状のＲＯＭはアドレス確定後のデータ出力のアクセスディレイにばらつきが多く、仕様としてはその最大遅延時間で規定されている．本文の中（(7.2.5)(1)）でも述べたがこのアクセスディレイのばらつきが小さいならばパイプライン化処理により時間をずらしながらHough変換演算，θ-ρヒストグラムの形成が可能となり速度は向上する．このようなＲＯＭが一般に出回るようになればHough変換演算速度は大幅に向上し処理速度のボトルネックとしてはθ-ρヒストグラム平面の形成に要する時間がクローズアップされてくる．この処理はθ-ρ平面メモリの読み出し，インクリメント，同一アドレスへの再書き込みとなっておりパイプライン化が最も困難な処理である．この点で第９章に示す原理的に根本的に異なる度数累積加算の方法が有効になって来ると考えられる．

第８章　連立漸化式による高速Hough変換

８．１緒言

Hough変換を実時間システムに応用するため高速化の様々な研究が行われている．本章ではHough変換原式(8.1)を連立漸化式を用いて逐次的に高速計算する手順について述べる．その概要については第６章で述べた．ここでは式(8.1)を厳密に表す連立漸化式を誘導し，この式からの近似式を導きその厳密解を示す．またこれをハードウエア化する上での検討を行う．この種の手法として初めて紹介された輿水等の方法（FIHT2)は次のとうりである．

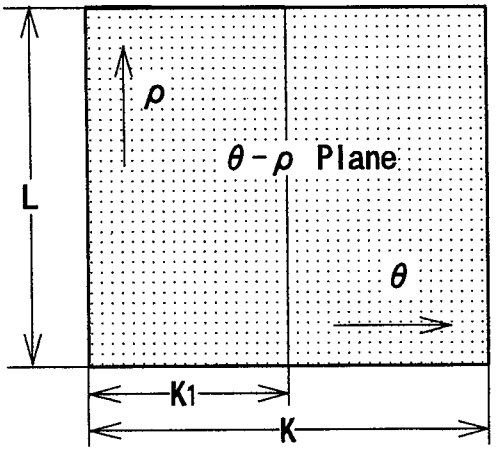


Fig.8.1 Dimension of K,K$_1$ and L on $\theta - \rho$ plane

$$\rho = x \cdot \cos\theta + y \cdot \sin\theta \qquad \text{---(8.1)}$$

図8.1に示すようにθ-ρヒストグラム平面のθ軸分割数を$K=[2^m \cdot \pi]$, $K_1=[2^{m-1} \cdot \pi]$, ρ軸分割数をLとし，ヒストグラム平面として大きさ$K \times L$の２次元配列を用意して，ρ値を次の式(8.2),(8.3)で順々に算出するものである．ρ_2は$n \cdot \Delta$が$\pi/2$だけずれた位置からのρの値である（図8.2参照）．ここで［　］は整数化を意味するガウス記号である．

$\rho_{1,0}=X$,　$\rho_{2,0}=y$,　$\Delta=1/2^m$

$$\rho_{1,n+1} = \rho_{1,n} + \Delta \cdot \rho_{2,n} \qquad \text{---(8.2)}$$

$$\rho_{2,n+1} = \rho_{2,n} - \Delta \cdot \rho_{1,n+1} \qquad \text{---(8.3)}$$

但し$\rho_{2,n}=\rho_{1,n+K1}$: $n=0,1,2,\ \cdots K_1-1$

この連立漸化式の一般解は穂坂により与えられており[63]

$$\rho_{1,n}=\{\cos(\Delta/2)\}^{-1} \cdot \{x \cdot \cos(n \cdot \Delta - \Delta/2) + y \cdot \sin(n \cdot \Delta)\} \qquad \text{---(8.4)}$$

$$\rho_{2,n}=\{\cos(\Delta/2)\}^{-1} \cdot \{-x \cdot \sin(n \cdot \Delta) + y \cdot \cos(n \cdot \Delta - \Delta/2)\} \qquad \text{---(8.5)}$$

となる．ここでΔは角度の微少変化量で，$\Delta=1/2^m \fallingdotseq 0$が満足されれば式(8.4),(8.5)は式(8.1)を近似する．また$n \cdot \Delta$は式(8.1)のθに相当する．この方法はパソコン，或いはワークステーションでの計算を想定すると，アセンブラで記述すればCPUレジスタ間のシフト演算（$\Delta=1/2^m$を乗じる演算がシフトに置き換えられる．）と加減算のみでρ値を逐次的に計算可能で処理速度も速い[23),(24),(28)]．そして占有するメモリ量も表参照方式等に比し非常に少ない．但し，式(8.4),(8.5)を見てわかるように角度について$\Delta/2$,振幅につ

いてcosの項が関係しているので式(8.1)とは若干のずれがあり，これは処理速度とのトレードオフの関係となっている．これを変形Hough変換として捉えれば誤差の問題は無くなる．すなわちθ-ρ平面からもとのX-Y平面上の直線に戻す式に補正項[24]を加えれば正確にその位置を特定可能である．

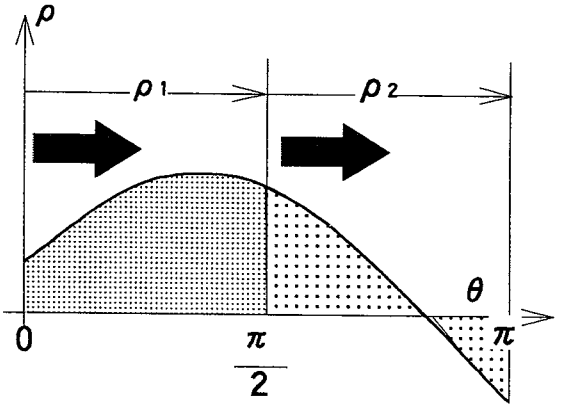


Fig.8.2 Covering range of ρ_1, ρ_2

しかし直線以外の他のパラメータ，例えばフェレ長，絶対最大長，周囲長等の測定については[1]，Hough曲線の包絡線を利用する関係で，正確な値を求めることには難がある．この意味でHough変換原式(8.1)に忠実で，且つ高速な漸化式があると好都合である．さらにFIHT2では式(8.2)で求めた$\rho_{1,n+1}$が式(8.3)右辺で使われており，処理上直列になっている．マルチプロセッサでの演算や，ハードウエア化について考えればその特長を生かし，並列性の高い漸化式が望ましい．本章ではこれらの漸化式の基本となる数学的に厳密な漸化式を導き（第８．２節），これまでに提案されてきた高速Hough変換の各種漸化式がこの厳密式から級数近似の打ち切り項数の取り方によってそれぞれ導出されることを示す（第８．３節）．そしてそれらの近似漸化式の一般解が行列の理論により求められることを示す（第８．４節）．次にこれらの式の形と一般解からの近似の程度により求められるρ値の精度について理論的な面から定性的に検討を加える（第８．５節）．第８．６節ではこれらの漸化式について計算機実験により精度の評価を行う．第８．７節ではこれらのアルゴリズムをハードウエアにインプリメントする場合の問題を取り扱う．第８．８節では８．２節で示した厳密式の係数行列が座標回転行列であることを利用し，係数行列として逆方向回転行列を掛けていくことにより４重並列にρ値が生成されることを示し，この４重並列化の副次的効果としてρ値の計算誤差が減少することを導く．そして並列化による高速なHough変換ハードウエアが構成可能なことを示す．各節の内容は常にアルゴリズムのハードウエア化を念頭に置き，随時これについての議論を差し挟んで話を進める．

８．２　厳密式の誘導

式(8.1)のρの領域をρ_1, ρ_2に分けて図8.2の如く領域分担させるとρ_2はρ_1と$\pi/2$ずれているので　$0 \leqq \theta < \pi/2$として

$$\rho_1 = x \cdot \cos\theta + y \cdot \sin\theta \qquad \text{---(8.6)}$$

$$\rho_2 = -x \cdot \sin\theta + y \cdot \cos\theta \qquad \text{---(8.7)}$$

この式を漸化式として捉え，Δを角度の微少変化量としてθを$n \cdot \Delta$と置けば第n項は

$$\rho_{1,n} = x \cdot \cos(n \cdot \Delta) + y \cdot \sin(n \cdot \Delta)$$

$$\rho_{2,n} = -x \cdot \sin(n \cdot \Delta) + y \cdot \cos(n \cdot \Delta)$$

そうすると第$n+1$項は三角関数の加法定理により

$$\rho_{1,n+1} = x \cdot \cos(n \cdot \Delta + \Delta) + y \cdot \sin(n \cdot \Delta + \Delta) = x \cdot \{\cos(n \cdot \Delta)\cos\Delta - \sin(n \cdot \Delta) \cdot \sin\Delta\} + y \cdot \{\sin(n \cdot \Delta) \cdot \cos(\Delta) + \cos(n \cdot \Delta) \cdot \sin\Delta\}$$

$$= \cos\Delta \cdot \rho_{1,n} + \sin\Delta \cdot \rho_{2,n}$$

$$\rho_{2,n+1} = -x \cdot \sin(n \cdot \Delta + \Delta) + y \cdot \cos(n \cdot \Delta + \Delta) = -x \cdot \{\sin(n \cdot \Delta) \cdot \cos\Delta + \cos(n \cdot \Delta) \cdot \sin\Delta\} + y \cdot \{\cos(n \cdot \Delta) \cdot \cos\Delta - \sin(n \cdot \Delta) \cdot \sin\Delta\}$$

$$= -\sin\Delta \cdot \rho_{1,n} + \cos\Delta \cdot \rho_{2,n}$$

対にして書き下すと

$$\rho_{1,n+1} = \cos\Delta \cdot \rho_{1,n} + \sin\Delta \cdot \rho_{2,n} \qquad \text{---(8.8)}$$

$$\rho_{2,n+1} = -\sin\Delta \cdot \rho_{1,n} + \cos\Delta \cdot \rho_{2,n} \qquad \text{---(8.9)}$$

行列形式で書けば

$$\rho_n = \begin{bmatrix} \rho_{1,n} \\ \rho_{2,n} \end{bmatrix} \quad \text{として}$$

$$\rho_{n+1} = \begin{bmatrix} \cos\Delta & \sin\Delta \\ -\sin\Delta & \cos\Delta \end{bmatrix} \cdot \rho_n = A \cdot \rho_n \qquad \text{---(8.10)}$$

但し初期値$(\rho_{1,0},\ \rho_{2,0})^T = (x, y)^T$とする．また$(x, y)$はHough変換演算を行おうとしているサンプリング点の座標値である．式(8,10)の係数行列AはA・A^T=Eであり，よく知られているように時計方向回りの座標回転行列である．

図8.3は初期値$(x, y)^T$から出発してρ_1，ρ_2が算出されていく過程を示している．■で示されている部分が接続点である．

ここで$\sin\Delta$，$\cos\Delta$を級数展開すると

$$\sin\Delta = \sum_{n=0}^{\infty} (-1)^n \frac{\Delta^{2n+1}}{(2n+1)!} = \Delta - \frac{\Delta^3}{3!} + \frac{\Delta^5}{5!} \quad \cdots\cdots \qquad \text{---(8.11)}$$

$$\cos\Delta = \sum_{n=0}^{\infty} (-1)^n \frac{\Delta^{2n}}{(2n)!} = 1 - \frac{\Delta^2}{2!} + \frac{\Delta^4}{4!} \quad \cdots\cdots \qquad \text{---(8.12)}$$

である．角度の変化量Δが微小なときに式(8.11),(8.12)右辺の第何項まで考慮するかによってこれまで取り上げられていた各種漸化式が導出され，その漸化式の精度が左右される．本節で得られた式(8.8),(8.9)を本節以降"厳密漸化式"と呼ぶ．この式は何等の近似も行っておらず，数学的にHough変換原式(8.1)を忠実に表現している．

8．3　近似漸化式の誘導

本節では厳密漸化式(8.10)の係数行列Aを構成する正弦，余弦関数の多項式近似の度合いにより，いくつかの漸化式が誘導されることを示す．議論する順序は式(8.11),(8,12)の項数を少なくとった順序(最もラフな近似の順）とする．

（1）　並列型漸化式の誘導

式(8.11),(8.12)右辺の第2項以下を無視すると式(8.10)は

$$\rho_{n+1}=\begin{bmatrix} 1 & \Delta \\ -\Delta & 1 \end{bmatrix}\cdot\rho_n \qquad \text{---(8.13)}$$

展開式では

$$\rho_{1,n+1}=\rho_{1,n}+\Delta\cdot\rho_{2,n} \qquad \text{---(8.14)}$$

$$\rho_{2,n+1}=-\Delta\cdot\rho_{1,n}+\rho_{2,n} \qquad \text{---(8.15)}$$

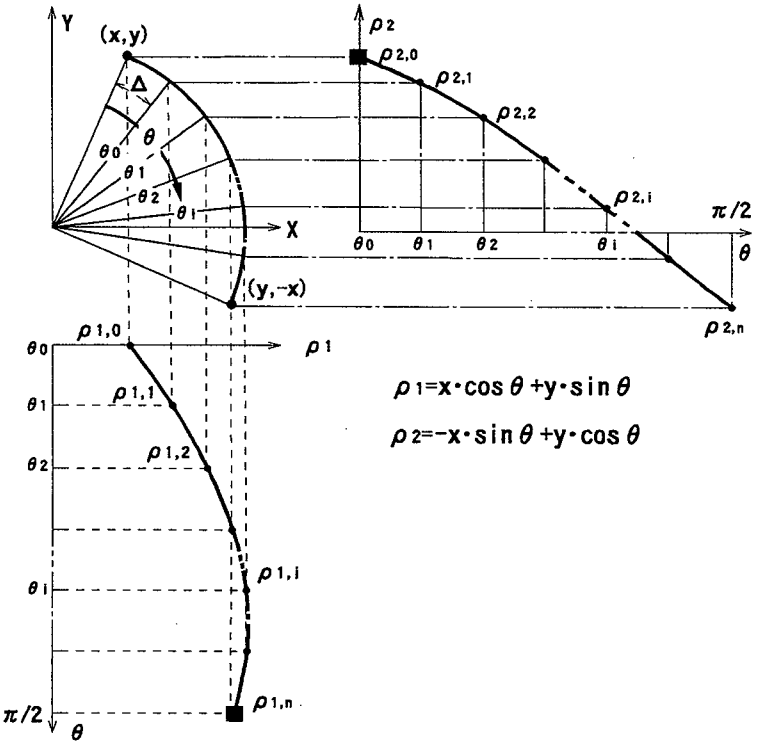


Fig.8.3 Calculation of ρ_1 and ρ_2 value using expression (8.8),(8.9)

ここでΔを$\Delta=2^{-m}$(mは正の整数)とおけばΔの乗算がビットシフト演算に置き換えられ1回のシフトと1回の加減算により一つのρ値の算出が可能となる．$\theta-\rho$ヒストグラム平面の寸法及びρ_1，ρ_2に与える初期値は8．1節で述べたとうりである．K，K_1の寸法も同様である．式(8.15)の右辺を見ると8．1節で取り上げた式(8.14)での結果ではなく前回の連立漸化式計算の結果が入っていることに注目しなければならない．つまり式(8.14)，(8.15)はそれぞれ独立したハードウェア上，或いは並列計算機上で1挙動で（ハードウェア上では1クロックで）演算が可能な形式になっている．

式(8.14),(8.15)は式(8.8),(8.9)の最もラフな近似であり，後で記すように精度は良くない．しかし高速性を備えている．式(8.13)を以後”並列型漸化式”と呼ぶことにする．

（2） ＦＩＨＴ２式の誘導

ここでは文献(24)に挙げられている輿水らのFIHT2を誘導する．これは8．1節で紹介したものである．まず式(8.11)の$sin\Delta$の近似を第1項で打ち切り，$sin\Delta\fallingdotseq\Delta$とする．次に式(8.12)の$cos\Delta$を一つは第1項で打ち切り$cos\Delta\fallingdotseq 1$，今一つは第2項で打ち切り$cos\Delta\fallingdotseq 1-\Delta^2/2$とし，更に$cos\Delta\fallingdotseq 1-\Delta^2$と近似する．そうすると

$$\rho_{n+1}=\begin{bmatrix} 1 & \Delta \\ -\Delta & (1-\Delta^2) \end{bmatrix}\cdot\rho_n \qquad \text{---(8.16)}$$

展開形で書けば

$\rho_{1,n+1} = \rho_{1,n} + \Delta \cdot \rho_{2,n}$

$\rho_{2,n+1} = -\Delta \cdot \rho_{1,n} + (1-\Delta^2)\rho_{2,n}$

さらに

$\rho_{1,n+1} = \rho_{1,n} + \Delta \cdot \rho_{2,n}$

$\rho_{2,n+1} = \rho_{2,n} - \Delta(\rho_{1,n} + \Delta \cdot \rho_{2,n})$

最後の式の右辺（　）の中は$\rho_{1,n+1}$に等しいから

$\rho_{1,n+1} = \rho_{1,n} + \Delta \cdot \rho_{2,n}$ ---(8.17)

$\rho_{2,n+1} = -\Delta \cdot \rho_{1,n+1} + \rho_{2,n}$ ---(8.18)

となりFIHT2の式が導出できた．以後式(8.16)を"FIHT2式"と呼ぶことにする．

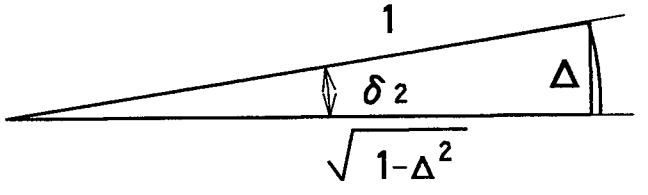


Fig.8.4 Relation between δ_2 and Δ

（３） 高精度漸化式の誘導

８．２節厳密漸化式の誘導で連立漸化式の係数行列Ａが時計方向回り、回転量Δの座標回転行列であることを示した．ここで図8.4に示すδ_2を導入すれば$\sin\delta_2 = \Delta$, $\cos\delta_2 = \sqrt{1-\Delta^2}$である．（ここでδの添字を2としたのは８．４節で述べる一般解の提示の順序に合わせたためである．）δ_2及びΔが非常に小さく$\delta_2 \fallingdotseq \Delta$とできることから係数行列Ａの座標回転量Δを$\delta_2$で置き換え

$$\rho_{n+1} = \begin{pmatrix} \cos\Delta & \sin\Delta \\ -\sin\Delta & \cos\Delta \end{pmatrix} = \begin{pmatrix} \cos\delta_2 & \sin\delta_2 \\ -\sin\delta_2 & \cos\delta_2 \end{pmatrix} = \begin{pmatrix} \sqrt{1-\Delta^2} & \Delta \\ -\Delta & \sqrt{1-\Delta^2} \end{pmatrix} \cdot \rho_n \quad \text{---(8.19)}$$

展開形では

$\rho_{1,n+1} = \sqrt{1-\Delta^2} \cdot \rho_{1,n} + \Delta \cdot \rho_{2,n}$ ---(8.20)

$\rho_{2,n+1} = -\Delta \cdot \rho_{1,n} + \sqrt{1-\Delta^2} \cdot \rho_{2,n}$ ---(8.21)

となる．この式は穂坂[63]によっても示されており，式(8.8),(8.9)を高精度に近似計算する式として有効であることが結論されている．事実、変数が単精度浮動小数点数で演算されるときその誤差は変数のビット長による有効桁数に近い近似が得られることを８．６節で述べる．但し式(8.20),(8.21)の√を含む項の乗算が、前記(1)並列型漸化式，(2)FIHT2式のように２のべき乗の性質を利用したシフト算に置き換え不可能なので処理速度の点で問題がある．以後式(8.19)を"高精度漸化式"と呼ぶことにする．

（４） 高速高精度漸化式の誘導

ここでは前３者の方法に比し近似の度合いを上げて，後で示すように実用上全くρ値計算の誤差が無視でき，且つ高速性を備えた近似式を誘導する．

式(8.11)の右辺第２項，,式(8.12)の右辺第３項以下を無視すれば

$$\rho_{n+1} = \begin{bmatrix} (1 - \frac{\Delta^2}{2}) & \Delta \\ -\Delta & (1 - \frac{\Delta^2}{2}) \end{bmatrix} \cdot \rho_n \quad \text{---(8.22)}$$

展開式で書けば

$$\rho_{1,n+1}=\left(1-\frac{\Delta^2}{2}\right)\cdot\rho_{1,n}+\Delta\cdot\rho_{2,n}=\rho_{1,n}-\frac{\Delta^2}{2}\cdot\rho_{1,n}+\Delta\cdot\rho_{2,n} \quad ---(8.23)$$

$$\rho_{2,n+1}=-\Delta\cdot\rho_{1,n}+\left(1-\frac{\Delta^2}{2}\right)\cdot\rho_{2,n}=-\Delta\cdot\rho_{1,n}+\rho_{2,n}-\frac{\Delta^2}{2}\cdot\rho_{2,n} \quad ---(8.24)$$

となる．これは文献(47)に示されている式である．ここでΔをΔ=2^{-m}（mは正の整数）とおいて乗算がシフト演算に置き換えられることに変わりはない．実際上の演算は2回のビットシフトと2回の加減算によりρ値1個の算出が可能となる．前3者の方法に比べて右辺の項数がそれぞれ3つなのでハードウエア化した場合，回路が多少複雑になる．以後式(8.22)を”高速高精度漸化式”と呼ぶことにする．

8．4　近似漸化式の一般解

この節では行列の理論[64]を用いて前節に挙げた4種の近似漸化式についてその一般解を導出した結果について記す．具体的な計算過程は付録としている．次節での考察のためまずHough変換原式、厳密漸化式から近似条件を含めて整理している．近似式の取り扱いの順序は前節の近似式の誘導のときと同じ順序とする．なお各式左辺のρ_{n+1}の添字は一般解の形に合わせて1つ減らしρ_nとしている．

Hough変換原式　$\rho=x\cdot\cos\theta+y\cdot\sin\theta$　---(8.25)

厳密漸化式　$\rho_n=\begin{bmatrix}\cos\Delta & \sin\Delta\\ -\sin\Delta & \cos\Delta\end{bmatrix}\cdot\rho_{n-1}$　---(8.26)

近似　近似なし

一般解　Hough変換原式そのもの

[a]並列型漸化式　$\rho_n=\begin{bmatrix}1 & \Delta\\ -\Delta & 1\end{bmatrix}\cdot\rho_{n-1}$　---(8.27)

近似　$\cos\Delta\fallingdotseq 1,\quad \sin\Delta\fallingdotseq\Delta$　---(8.28)

一般解　$\rho_{1,n}=(\sqrt{1+\Delta^2})^n\cdot\{x\cdot\cos(n\cdot\delta_1)+y\cdot\sin(n\cdot\delta_1)\}$　---(8.29)

$\rho_{2,n}=(\sqrt{1+\Delta^2})^n\cdot\{-x\cdot\sin(n\cdot\delta_1)+y\cdot\cos(n\cdot\delta_1)\}$　---(8.30)

但しδ_1は$\delta_1=\tan^{-1}\Delta$である．

[b]ＦＩＨＴ２式　$\rho_n=\begin{bmatrix}1 & \Delta\\ -\Delta & (1-\Delta^2)\end{bmatrix}\cdot\rho_{n-1}$　---(8.31)

近似　$\cos\Delta\fallingdotseq 1$及び$\cos\Delta\fallingdotseq 1-\Delta^2$，$\sin\Delta\fallingdotseq\Delta$　---(8.32)

一般解　$\rho_{1,n}=\{\cos(\Delta/2)\}^{-1}\cdot\{x\cdot\cos(n\cdot\Delta+\Delta/2)+y\cdot\sin(n\cdot\Delta)\}$

$=\{\cos(\Delta/2)\}^{-1}\cdot\{x\cdot\cos(n\Delta)\cos(\Delta/2)+y\cdot\sin(n\Delta)+x\cdot\sin(n\Delta)\cdot\sin(\Delta/2)\}$　---(8.33)

$\rho_{2,n}=\{\cos(\Delta/2)\}^{-1}\cdot\{-x\cdot\sin(n\cdot\Delta)+y\cdot\cos(n\cdot\Delta+\Delta/2)\}$

$=\{\cos(\Delta/2)\}^{-1}\cdot\{-x\cdot\sin(n\Delta)+y\cdot\cos(n\cdot\Delta)\cdot\cos(\Delta/2)+y\cdot\sin(n\cdot\Delta)\cdot\sin(\Delta/2)\}$　---(8.34)

[c]高精度漸化式　$\rho_n=\begin{pmatrix}\sqrt{1-\Delta^2} & \Delta\\ -\Delta & \sqrt{1-\Delta^2}\end{pmatrix}\cdot\rho_{n-1}$　---(8.35)

近似　　$\delta_2 \fallingdotseq \Delta$，但し図8.4の関係を満足するものとする．

一般解　　$\rho_{1,n} = x \cdot cos(n \cdot \delta_2) + y \cdot sin(n \cdot \delta_2)$　　---(8.36)

$\rho_{2,n} = -x \cdot sin(n \cdot \delta_2) + y \cdot cos(n \cdot \delta_2)$　　---(8.37)

[d]高速高精度漸化式　$\rho_n = \begin{bmatrix} (1 - \frac{\Delta^2}{2}) & \Delta \\ -\Delta & (1 - \frac{\Delta^2}{2}) \end{bmatrix} \cdot \rho_{n-1}$　　---(8.38)

近似　　$cos\Delta \fallingdotseq 1 - \Delta^2/2, \quad sin\Delta \fallingdotseq \Delta$

一般解　　$\rho_{1,n} = (\sqrt{1 + \Delta^4/4})^n \cdot \{ x \cdot cos(n \cdot \delta_3) + y \cdot sin(n \cdot \delta_3) \}$　　---(8.39)

$\rho_{2,n} = (\sqrt{1 + \Delta^4/4})^n \cdot \{ -x \cdot sin(n \cdot \delta_3) + y \cdot cos(n \cdot \delta_3) \}$　　---(8.40)

但しδ_3は$\delta_3 = tan^{-1}\{\Delta/(1-\Delta^2/2)\}$である．

8．5　近似漸化式の精度及び速度

本項ではこれまでの節で厳密漸化式から近似によって導いた４つの漸化式について近似度と一般解の形からρ値の計算精度を推定する定性的な考察を行う．

8．5．1　精度について

”近似”と記している項目は$cos\Delta$，$sin\Delta$のべき級数展開における項の第何項までとるのか，或いは図8.4のδ_2とΔの関係を示している．これは[a]並列型漸化式，[b]FIHT2式，[d]高速高精度漸化式の順に項数を多くとっている．従って直感的にも精度はこの順に向上すると考えられる．但し[b]FIHT2式に関しては$cos\Delta$の第２項までの展開形は$cos\Delta = 1 - \Delta^2/2$であるがこれを$cos\Delta = 1 - \Delta^2$としている点に問題が残っている．しかし[a]並列漸化式の$cos\Delta = 1$よりは近似度が良いことは明確である．また[c]高精度漸化式は他の式と近似の形式が異なっている．それぞれの”一般解”について見れば｛　｝の外側に掛かる係数が正弦関数の振幅方向の誤差をもたらしている．[a]並列漸化式と[d]高速高精度漸化式を比較して見れば[a]並列漸化式のΔ^2が[d]高速高精度漸化式では$\Delta^4/4$となっており，$\Delta = 2^{-m}$のmが比較的実用的な範囲の値 $m=7$をとると，このときnの最大値は$n \fallingdotseq 200$となるが[a]並列漸化式では $(1 + \Delta^2)^{200/2} \fallingdotseq 1 + 100 \cdot \Delta^2 = 1.0061$となる．[d]高速高精度漸化式では $(1 + \Delta^4/4)^{200/2} \fallingdotseq 1 + 25 \cdot \Delta^4 = 1.000000093$となり[d]高速高精度漸化式の方がはるかに小さい値となる．この係数項はρ_1とρ_2が$\pi/2$の位置において接続する時に接続点で段差を生じる原因となる．一方[b]FIHT2式については振幅方向の誤差 $\{cos(\Delta/2)\}^{-1}$はnに無関係に一定でρ_1，ρ_2の接続点における段差は比較的少ないと考えられる．ちなみに$m=7$の時 $\{cos(\Delta/2)\}^{-1} = 1.0000076$となる．但し[b]FIHT2式には[a]並列漸化式，[d]高速高精度漸化式と異なり式(8.33),(8.34)でcosの項の中に$\Delta/2$の位相のずれがあり，これが角度方向の誤差を発生させる要因となることが予想される．[c]高精度漸化式の誤差は式の形からだけでは予測し難くδ_2とΔの差異を定量的に計算しなければならない．（8.6.1項参照）

8．5．2　実行速度について

実行速度を問題にする場合，そのアルゴリズムが現実の装置にどのような形でインプリメントされるかが非常に重要である．ここでは①シングルCPUと，②並列型CPUまたはハー

ドウエア，の２つの場合に分けて考える．両者の決定的な違いは処理の並列実行の可能性である．後者ではこれが可能で前者では不可能である．

①シングルCPUで実行する場合

まず①の立場から見れば次のようになる．式(8.27),(8.31),(8.35).(8.38)の順で係数行列が複雑になっている．式(8.35)を除く３つの式はいずれも乗算のシフト演算への変換が可能な式である．従ってこの順に速度は遅くなると推測されるが[b]FIHT2式のオリジナル式(8.2),(8.3)を見ると式(8.2)の結果が式(8.3)で用いられているだけで，一どきに１つの処理しか行わない①の仮定から[a]並列漸化式と同じ処理速度であると考えられる．[d]高速高精度漸化式の場合，その展開形式(8.23),(8.24)から判るが右辺の項数が多い分遅くなる．同一ビット長の四則演算において，演算速度は遅い順に除算，乗算，減算，加算，シフト算となる．ここで，各漸化式について演算処理の種類と回数を考える．いずれも$\pi/2$の位置で接続される一組のρ値(ρ_1，ρ_2)を算出するに要する時間数を示している．なお次の結果はそれぞれ上掲の漸化式の展開形で見る必要がある．つまり[a]並列漸化式は式(8.14),(8.15)，[b]FIHT2式は式(8.17),(8.18)，[c]高精度漸化式は式(8.20),(8.21)，[d]高速高精度漸化式は式(8.23),(8.24)である．

[a]並列漸化式はmビットシフト算2回と加減算各１回に相当する時間．[b]FIHT2式も同じ．[c]高精度漸化式は乗算２回とmビットシフト２回，加減算各１回に相当する時間．($\sqrt{1-\Delta^2}$の値は予め計算して準備しておけばよいから)

[d]高速高精度漸化式は$2m+1$ビットシフト算2回とmビットシフト2回，加減算4回に相当する時間．

[c]高精度漸化式については乗算のシフト算への置き換えが不可能である．しかし今般のワークステーション或いは高級パソコンがハードウエアでの高速演算機構を標準装備している現状を勘案すると極端な処理速度の低下はないであろう．ある特別な例では浮動小数点乗算の方が固定小数点乗算よりも高速であることが確認されている．従って，速度的には[a]並列漸化式，[b]FIHT2式が同程度，その後[d]高速高精度漸化式，[c]高精度漸化式の順であろうと考えられる．最後に述べた特別な場合には[c]高精度漸化式と[d]高速高精度漸化式が逆転することもあり得る．

②並列型CPUまたはハードウエア上で実行する場合

次に②の立場で考察する．[b]FIHT2式では式(8.31)からみると並列演算の形になっているがオリジナル式(8.2),(8.3)を見ると演算が直列になっているので[a]並列型漸化式の２倍の時間を要する．或いは式(8.31)で直接計算すれば$(1-\Delta^2)$の乗算が[a]並列型漸化式より手間がかかる分だけ遅くなる．また[c]高精度漸化式は$\sqrt{1-\Delta^2}$の乗算が２のべき乗化できないので更に遅いと推定される．ハードウエア化した場合についてはシフト演算は単なる線のつなぎ変えだけだから必要時間は0である．ここで一組のρ値(ρ_1，ρ_2)を算出するに要する最大時間についてまとめると次の様になる．

[a]並列漸化式　並列に実行できるから，加減算の組み合わせ論理回路の遅延時間に相当する1クロック時間．

[b]FIHT2式　直列であり組み合わせ論理回路だから回路の遅延時間に相当するクロック時間２個分．

[c]高精度漸化式　並列に実行できるから乗算１回と加減算１回分の回路の遅延時間に相当する１クロック時間.

[d]高速高精度漸化式　並列に実行でき且つ組み合わせ論理回路だから回路の遅延時間に相当する１クロック時間.

結論として高速な順に示せば[a]並列漸化式，[b]FIHT2式,[d]高速高精度漸化式，[c]高精度漸化式の順となる.

表8.1に以上のことを考慮してまとめている．表中◎，○，△，×は有意差が２倍以上のときにその使い分けをしている．表中必要最低ビット長はシフト加算の結果がアンダーフローしないビット数の限界である．これについては８．７節で議論する．[c]高精度漸化式では浮動小数点数として捉え，単精度浮動小数点数の一般的な長さ32ビットとしている.

Table 8.1 Comparison of 4 method
Assumed　m=7,　K=402

	[a] Parallel	[b] FIHT2	[c] High-precision	[d] High-speed & High-precision
Accuracy	△	△	◎	◎
Bit Length Min.	18	18	32	26
Speed on hardware	○	△	×	○
Speed on software	○	○	△	○

８．６　ソフトウエアシミュレーション

本節では，これまでの節で述べた４つの連立漸化式について，誤差や演算速度等を計算機実験の結果と比べながら評価する.

８．６．１　各方式における最大誤差の推定

本項では前節までに示した各漸化式の誤差の推定を行う．それぞれ漸化式の一般解の各変数に誤差が最大となる場合の値を代入してその値を見積もる．ここでは軸分割数が比較的実用的な値となるm=7を仮定し，$\Delta=2^{-7}=1/128=7.8125000000\times10^{-3}$, K=402, K_1=201として議論する．X-Y平面のx, yの値域は$0\leqq x, y<512$とする．本項以降このように考える.

[a]並列漸化式の場合

一般解の式(8.29),(8.30)のδ_1は$\Delta=1/128$より計算すると$\delta_1=7.8123410601\times10^{-3}$である．この時$n$=201に対して$|n(\Delta-\delta_1)/\Delta|=4.09\times10^{-3}$であるから$\delta_1$は$\Delta$で十分置き換え可能である．計算式が漸化式であるため誤差はnの増大と共に累積するからnの最大値n=201で最大誤差が発生する．この式右辺の2番目の項についてn=201,すなわち$n\Delta=\pi/2$近傍での真値との誤差ε_aは式(8.29)より

$$\varepsilon_a=\{(\sqrt{1+\Delta^2})^n-1\}\cdot\max.\rho_n\fallingdotseq n/2\cdot\Delta^2\cdot\max.\rho_n=n/2^{15}\cdot\max.\rho_n$$

n=201として，max.ρ_{201}=511=2^9-1なので

$\varepsilon_a \fallingdotseq 201/2^6 \fallingdotseq 3.14$[dot]　となる．

[b]ＦＩＨＴ２式の場合

この方式の誤差は(8.33),(8.34)式の最終項$x\cdot sin(n\Delta)\cdot sin(\Delta/2)$または $y\cdot sin(n\Delta)\cdot sin(\Delta/2)$が誤差の最大発生要因である．前述と同じく誤差の最大値はnが最大の位置で発生する．最大誤差ε_bはx=511,n=201,$\Delta=2^{-7}$を代入して

$\varepsilon_b=1/\{cos(1/256)\}\cdot 511\cdot sin(201/128)\cdot sin(1/256) \fallingdotseq 2.00$　[dot]

となる．単位[dot]は各平面の分割の最小単位である．

[c]高精度演算方式の場合

一般解の式(8.36),(8.37)のδ_2はΔ=1/128より計算するとδ_2=7.8125794750×10^{-3}である．この時n=201に対して$|n(\Delta-\delta_2)/\Delta|$=2.03×$10^{-3}$であるから$\delta_2$は$\Delta$で十分置き換え可能である．この置き換えをすれば式(8.36)，(8.37)はHough変換原式(8.8),(8.9)と一致するから本方式[ｃ]での誤差ε_cは

$\varepsilon_c \fallingdotseq 0$[dot]　である

[d]高速高精度漸化式の場合

一般解の式(8.39),(8.40)のδ_3はΔ=1/128より計算するとδ_3=7.8125794714×10^{-3}である．この時n=201に対して$|n(\Delta-\delta_3)/\Delta|$=2.02×$10^{-3}$であるから$\delta_3$は$\Delta$で十分置き換え可能である．[ａ]の場合と同じく最大誤差はnが最大の位置で発生する．この式右辺の2番目の項についてn=201,すなわち$n\Delta=\pi/2$近傍での真値との誤差ε_dは式(8.39)より

$\varepsilon_d=\{(\sqrt{1+\Delta^4/4})^n-1\}\cdot\max.\rho_n \fallingdotseq n/2\cdot\Delta^4/4\cdot\max.\rho_n=n/2^{31}\cdot\max.\rho_n$

[a]と同じn，max.ρ_nを代入すると

$\varepsilon_d \fallingdotseq n/2^{22}=201/4,194,304 \fallingdotseq 4.79\times10^{-5}$　[dot]

となりほとんど無視できる．

このように[ｃ]高精度漸化式及び[ｄ]高速高精度漸化式は極めて誤差が少ないことが判る．

８．６．２　ソフトウエアシミュレーションによる誤差の比較

各漸化式によって求めたρ値とHough変換原式(8.1)によって求めた真値との誤差の最大値は前節の議論によりX-Y平面の右端上部で発生する．この点x=511,y=511について，各漸化式による角度θにおけるρ値の真値（式(8.1)]に対する，漸化式を用いて計算した値との誤差を求めた．

図８．５には上段[ａ]並列型漸化式，下段[ｂ]ＦＩＨＴ２式について誤差を縦軸，θを横軸として示している．最大誤差はρ_1とρ_2の接続点，$\pi/2$で現れ[ａ]並列漸化式で3.247[dot]，[ｂ]ＦＩＨＴ２式で2.00[dot]であった．[ｃ]高精度演算方式，[ｄ]高速高精度漸化式については最大誤差がそれぞれ0.0038[dot]，0.0076[dot]と計算された．図上での相対比較では誤差０とみなして図の表示は省略した．使用した変数は単精度浮動小数点数である．上段[ａ]並列型漸化式，下段[ｂ]ＦＩＨＴ２式での誤差の最大値は前項で推定した値とほぼ一致していることがわかる．[ｃ]，[ｄ]での計算誤差は取り扱っているx,yの値が500前後であるので，真値として採用したコンパイラの組込関数であるcos,sinの誤差に近い値であり，誤差は全く無視できると言ってよい．

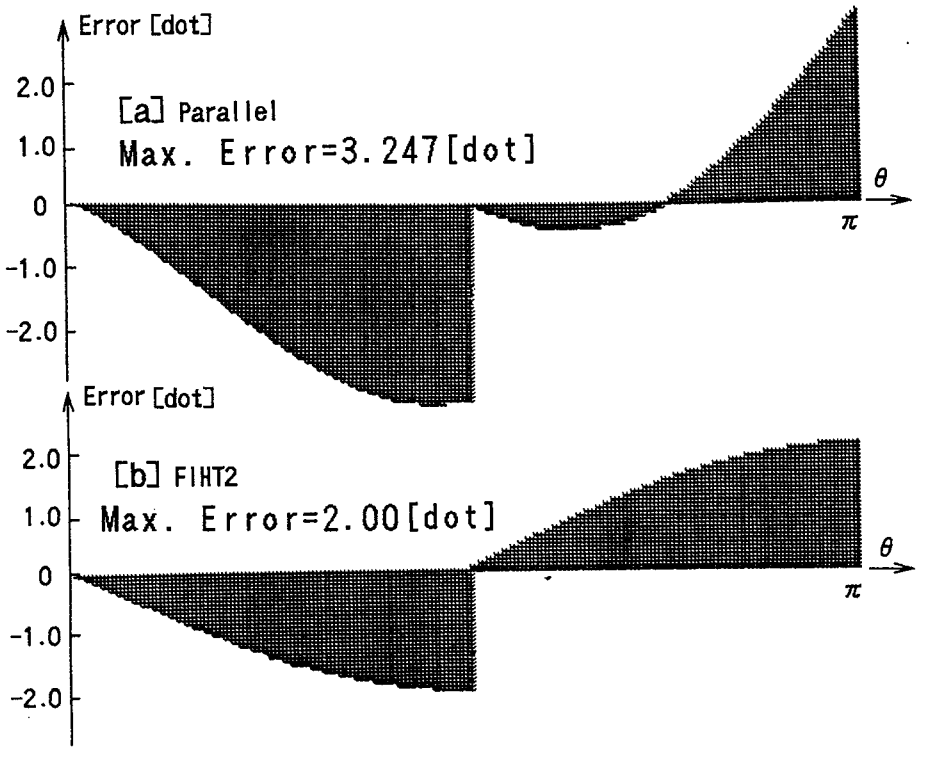


Fig.8.5 Caluculative error of each method

8．6．3　処理速度の比較

8．5．2項①の仮定（シングルCPU）のもとで速度についての計算機実験を行った結果を表8．2に記す．処理時間はθ-ρヒストグラム平面の1点をHough変換演算を行うのに要する時間である．原図形のサンプリング点1個当たりの時間に直すならθ軸分割数を乗じてやればよい．

Table 8.2 Comparison of processing speed

	Processing time for 1 point on θ-ρ plane
[a]Parallel	1 μ s
[b]FIHT2	1 μ s
[c]High-precision	2．1 μ s
[d]High-speed & High-precision	1．8 μ s

この結果は順序的に8．5．2項での予想に一致している．[c]が意外と遅くない結果が出ている．しかしこれは後で述べるハードウエア化の際に乗算部を用意しなければなら

ないため単純に好結果と見ることはできない．精度の点では[d]で十分でありハードウエア化も容易なことから高精度が要求される応用なら[d]，速度が問題ならハードウエア化のことも考慮して[a]が適当と考えられる．なおこの計算機実験の環境は計算機がPC9801FA(CPU i80486DX4 66MHz),使用した言語は"Turbo C V.2.0" である．

8．7 漸化式のハードウエア構成

前節までに述べてきた各種漸化式を実時間システムに応用するためにはハードウエア化が必須である．例えば表8．2の結果（ソフトウエア処理）から，サンプリング点の数が800［点］程度とすると1［μs］×402（軸分割数）×800（サンプリング点の数）≒0.32[s]必要で空間微分，度数探索その他の処理時間を合計すると秒のオーダーになる．本節では手順のハードウエア化について考察するが[c]については取り扱わない．精度が同等で速度，回路の複雑さの点で[d]に劣るからである．取り扱う数値はハードウエア化の容易性を考慮し，符号つき固定小数点形式で，小数点以下のビットを有するものとする．X-Y平面上のサンプリング点が最大値x=511，y=511の値をとったとき，ρ値の最大値は$\sqrt{2}\times 511$≒723となり，$2^{10}=1024>723>512=2^{9}$であるから$\rho$値の整数部は，符号ビットを含めて11ビット必要である．小数部は[a]，[b]と[d]では異なる．[a]，[b]では式(8.14),(8.15),(8.17),(8.18)よりシフト量がm=7ビットなので図8．6(a)より7ビット必要である．[d]では式(8.23),(8.24)よりシフト量が$2m$+1=15であるから図8．6(b)より15ビットである．まとめるとm=7のとき[a]，[b]では最低必要ビット長は合計18ビット[c]では最低必要ビット長26ビットとなる．但し最低必要ビット長とは8．6．2項での結果としての精度を維持するに最低必要な（十分ではないかもしれない）ビット長である（8．8．2項で述べるように必要且つ十分なビット長はこれより少し多い）．ソフトウエアでの処理ではアクセスが1バイト単位(8ビット)となるので[a]，[b]では24ビット，[d]では32ビットとなる．逆に考えると[d]は精度が高いのだからビット数が多いのは当然とも言える．[c]ではIEEEの単精度浮動小数点数の規格に準拠すれば32ビットとなる．

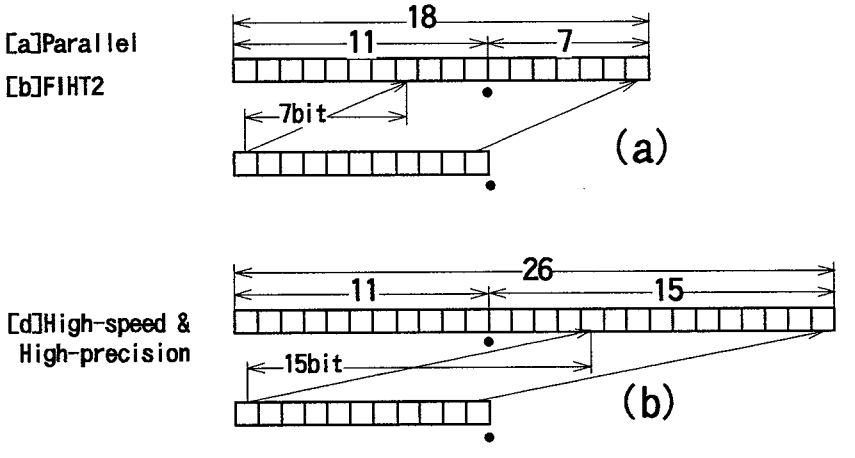


Fig.8.6 Relation between amount of shift and total bit length

（1）［a］並列漸化式

並列漸化式の手順をハードウエア化すると図8．7のようになる．まず最初にスイッチ

が(x, y)側に切り替えられて初期値をρ1，ρ2ラッチにセットする．このラッチはエッジトリガタイプである．

図中 Sift m bit の部分はブロックには表現しているが実際には単なるビット線のつなぎ換えで済んでいる．ρ1の出力はmビットのシフトを受けてρ2側の減算器のマイナス側に入る．

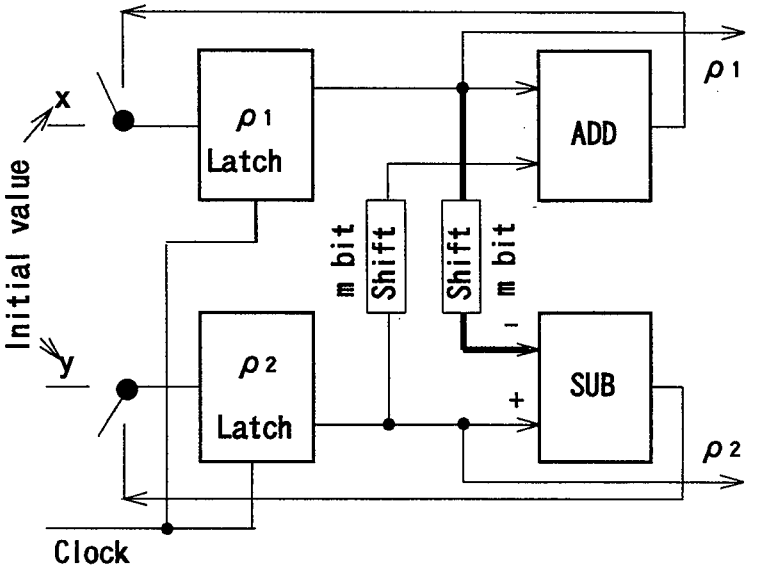


Fig.8.7 Block diagram of [a] Paralell method

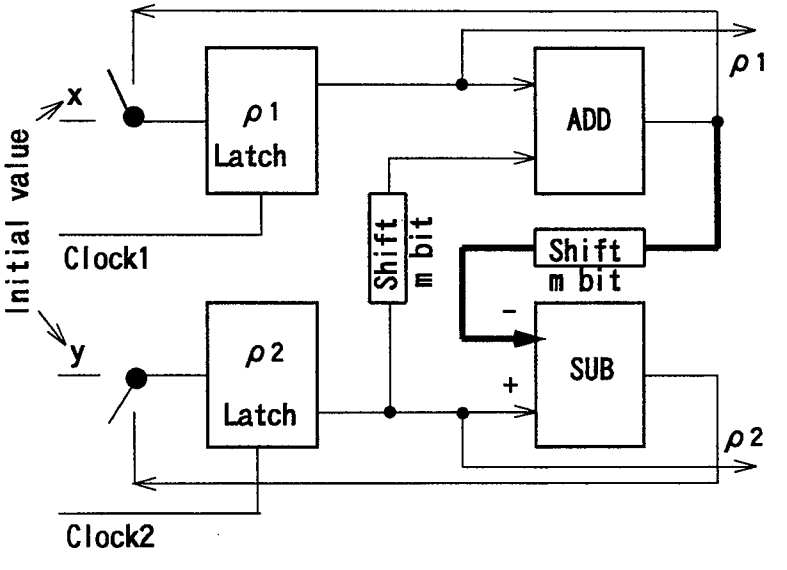


Fig.8.8 Block diagram of [b] FIHT2

その後クロックパルスの入力によりρ1，ρ2が順次生成されていく．ρ値はθ-ρ平面のρ側アドレスとして入力されるので小数点以下は切り捨てられる．

（2）［b］ＦＩＨＴ２式

ＦＩＨＴ２式の手順をハードウエア化すると図８．８のようになる．基本的に[a]並列

漸化式の場合と同じであるが図８．７と図８．８で太線で示している部分が異なる．これはこの手順が直列演算の形になっているからである．またそのために位相のずれた２種類のクロックを必要とする．

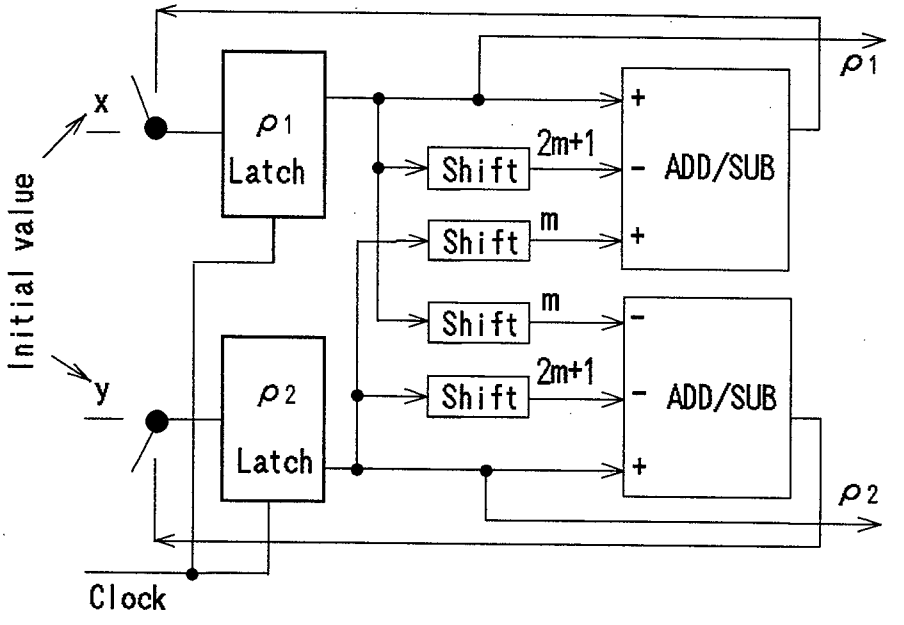


Fig.8.9 Block diagram of [d] High−speed & High−precision method

（３）［d］高速高精度漸化式

高速高精度漸化式の手順をハードウエア化すると図８．９のようになる．この場合も前２者と同じような形のブロック図になる．大きく異なるのはADD/SUBで示された加算／減算器の部分である．この部分は本来，加算器２個と減算器２個で構成するの普通であるがＰＧＡ(Programable Gate Array)を用いて一挙に加減算を処理している．このようにすることによって回路の簡素化と速度の向上が図れる．この回路の遅延時間がクロックレート（演算速度）を決定する．

８．８　多重並列漸化式によるHough変換(特許2)

本節では８．２節で示した座標回転行列をもとに時計方向と反時計方向に回転させる２種類の座標回転行列を用意する事によって一つの初期値（実質的には２個）から出発して回転角がπ/4の間で，ρ曲線一本分のρ値が算出できることを示す．この座標回転を時計方向，反時計方向に回転させながらρ値を算出する原理は前節までのどの漸化式にも適用可能である．この原理によれば４元連立漸化式（厳密には２元連立漸化式を２個用意する）から次々とρ値が同時並列に算出できる．なおこの式はπ/4, 3π/4におけるρの初期値を与えてやれば更に８重或いは16重に拡張できるので以後これを”多重並列漸化式”と呼ぶ．前節までに述べた漸化式の中で[a]並列漸化式及び[b]FIHT2は高速性はあるがハフ変換原式(8.1)に対してρ値に誤差が発生する．この誤差曲線を描くと図８．５に示したようにπ／２の位置で段差を生じる．この現象は元の直線がX軸に平行な時に，対応するθ-ρ平面のピーク点位置が分離するという結果を招く．ところが時計方向と反時計方向からのρ

値を算出する厳密漸化式の初期値はθ軸方向0，π／2，πから与えられるのでこの段差が消滅し円滑なハフプロットが可能になる．そこで本節では最も高速だが若干の誤差を示す[a]並列漸化式にこの原理を適用して高速性と精度向上の両立を目指す．

８．８．１　多重並列化の原理

まず８．２節で挙げた厳密漸化式についての議論を簡単に再掲する．

式(8.1)のρの領域をρ_1, ρ_2に分けて図8.2の如く領域分担させるとρ_2はρ_1とπ/2ずれているので　$0 \leqq \theta < \pi/2$として

$$\rho_1 = x \cdot \cos\theta + y \cdot \sin\theta \qquad \text{---(8.43)}$$

$$\rho_2 = -x \cdot \sin\theta + y \cdot \cos\theta \qquad \text{---(8.44)}$$

この２式から加法定理を用いて

$$\rho_n = \begin{bmatrix} \rho_{1,n} \\ \rho_{2,n} \end{bmatrix}$$

と置くと

$$\rho_{n+1} = \begin{bmatrix} \cos\Delta & \sin\Delta \\ -\sin\Delta & \cos\Delta \end{bmatrix} \rho_n = A\,\rho_n \qquad \text{---(8.45)}$$

ここで

$$A = \begin{bmatrix} \cos\Delta & \sin\Delta \\ -\sin\Delta & \cos\Delta \end{bmatrix}$$

AはA・A^T＝［Ｅ］であり，良く知られているようにΔを正にとれば座標値($\rho_{1,n}$, $\rho_{2,n}$)を時計方向に回転させる座標回転行列である．この座標回転の様子は図８．３に示されている．

図８．３を参照すると(x, y)なる初期値が$-\pi/2$なるθ方向の回転を受けて$(y, -x)$になっている．これを利用して$(y, -x)$なる位置から逆方向（反時計回り）に回転行列を乗じることによりθ＝π/4までのρ値が求まるはずである．そこで式(8.43),(8.44)をπ/2位相を進めれば

$$\rho_{2,1} = -x \cdot \sin\theta + y \cdot \cos\theta \qquad \text{---(8.46)}$$

$$\rho_{2,2} = -x \cdot \cos\theta - y \cdot \sin\theta \qquad \text{---(8.47)}$$

となる．８．２節の時計方向回りで生成されるρと区別するためρの第１添字を２としている．この式のθ＝0に於ける値は$(\rho_{2,1,0}, \rho_{2,2,0})^T = (y, -x)^T$である．ここでθ＝$-n \cdot \Delta$として第$n$項は

$$\rho_{2,1,n} = x \cdot \sin(n \cdot \Delta) + y \cdot \cos(n \cdot \Delta) \qquad \text{---(8.48)}$$

$$\rho_{2,2,n} = -x \cdot \cos(n \cdot \Delta) + y \cdot \sin(n \cdot \Delta) \qquad \text{---(8.49)}$$

である．式(8.48)の漸化式の項$\rho_{2,1,n+1}$はその前項によって

$$\begin{aligned} \rho_{2,1,n+1} &= x \cdot \sin(n \cdot \Delta + \Delta) + y \cdot \cos(n \cdot \Delta + \Delta) \\ &= x\{\sin(n \cdot \Delta) \cdot \cos(\Delta) + \cos(n \cdot \Delta) \cdot \sin(\Delta)\} \\ &\quad + y\{\cos(n \cdot \Delta)\cos(\Delta) - \sin(n \cdot \Delta)\sin(\Delta)\} \\ &= \cos\Delta \cdot \rho_{2,1,n} - \sin\Delta \cdot \rho_{2,2,n} \end{aligned}$$

と表される．また式(8.49)の漸化式の項$\rho_{2,2,n+1}$はその前項により

$$\begin{aligned} \rho_{2,2,n+1} &= -x \cdot \cos(n \cdot \Delta + \Delta) + y \cdot \sin(n \cdot \Delta + \Delta) \\ &= -x\{\cos(n \cdot \Delta) \cdot \cos(\Delta) - \sin(n \cdot \Delta) \cdot \sin(\Delta)\} \end{aligned}$$

$$+y\{sin(n\cdot\Delta)cos(\Delta)+cos(n\cdot\Delta)sin(\Delta)\}$$

$$=sin\Delta\cdot\rho_{2,1,n}+cos\Delta\cdot\rho_{2,2,n}$$

と表される．上式二つを整理して

$$\rho_{2,1,n+1}=cos\Delta\cdot\rho_{2,1,n}-sin\Delta\cdot\rho_{2,2,n} \qquad ---(8.50)$$

$$\rho_{2,2,n+1}=sin\Delta\cdot\rho_{2,1,n}+cos\Delta\cdot\rho_{2,2,n} \qquad ---(8.51)$$

但し初期値$(\rho_{2,1,0},\ \rho_{2,2,0})^T=(y,-x)^T$である．ここで

$$\rho_{2,n}=\begin{bmatrix}\rho_{2,1,n}\\ \rho_{2,2,n}\end{bmatrix}$$

$$\rho_{2,n+1}=\begin{bmatrix}cos\Delta & -sin\Delta\\ sin\Delta & cos\Delta\end{bmatrix}\rho_{2,n}=B\,\rho_{2,n} \qquad ---(8.52)$$

$$B=\begin{bmatrix}cos\Delta & -sin\Delta\\ sin\Delta & cos\Delta\end{bmatrix}$$

Bは$B\cdot B^T=E$であり，また$B=A^T$である．Bは座標$(\rho_{2,1,n},\ \rho_{2,2,n})$を反時計方向に回転させる座標回転行列となっていることがわかる．ここで８．２節に述べた時計方向回転によるρ値の式(8.8),(8.9)についてρの添字を１として書き改めれば

$$\rho_{1,1,n+1}=cos\Delta\cdot\rho_{1,1,n}+sin\Delta\cdot\rho_{1,2,n} \qquad ---(8.53)$$

$$\rho_{1,2,n+1}=-sin\Delta\cdot\rho_{1,1,n}+cos\Delta\cdot\rho_{1,2,n} \qquad ---(8.54)$$

但し初期値$(\rho_{1,1,0},\ \rho_{1,2,0})^T=(x,y)^T$とする．ここで

$\rho_{1,n}=\begin{bmatrix}\rho_{1,1,n}\\ \rho_{1,2,n}\end{bmatrix}$　　と置くと

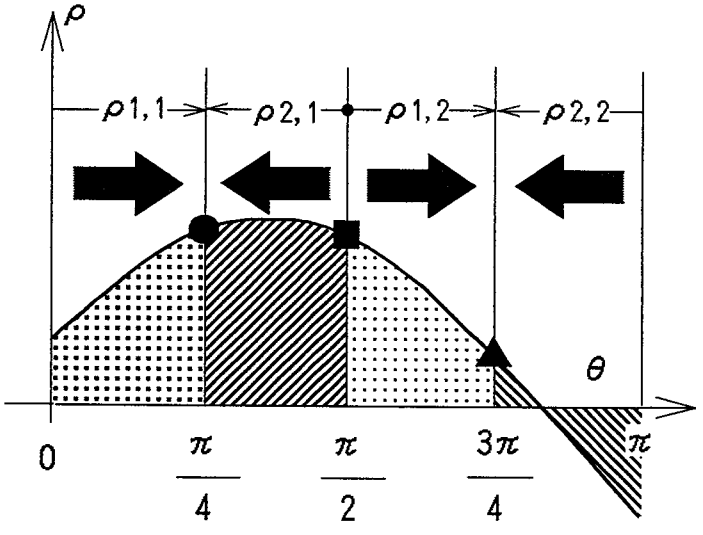


Fig.8.10 Covering range of ρ_{11}, ρ_{12}, ρ_{21} and ρ_{22}

$$\rho_{1,n+1}=\begin{bmatrix}cos\Delta & sin\Delta\\ -sin\Delta & cos\Delta\end{bmatrix}\rho_{1,n}=A\,\rho_{1,n} \qquad ---(8.55)$$

ここで

$$A=\begin{bmatrix}cos\Delta & sin\Delta\\ -sin\Delta & cos\Delta\end{bmatrix}$$

図８．２に対応させてρの分担領域を描けば図８．１０のようになる．図中の矢印はρが生成される順序方向を表している．

結局，初期値$(x, y)^T$から$\rho_{1,1}$，$\rho_{1,2}$が，初期値$(y, -x)^T$から$\rho_{2,1}$，$\rho_{2,2}$が順次生成されることがわかった．この生成の模様を図８．３に対応させて描けば図８．１１のようになる．ここで記号●，■，▲の種類は各ρの接続点を表し図８．１０，図８．１１のそれぞれの記号と対応している．Δが微少なとき$\cos\Delta \fallingdotseq 1$，$\sin\Delta \fallingdotseq \Delta$と近似すると（[a]並列漸化式の場合と同じ近似）式(8.53),(8.54)より

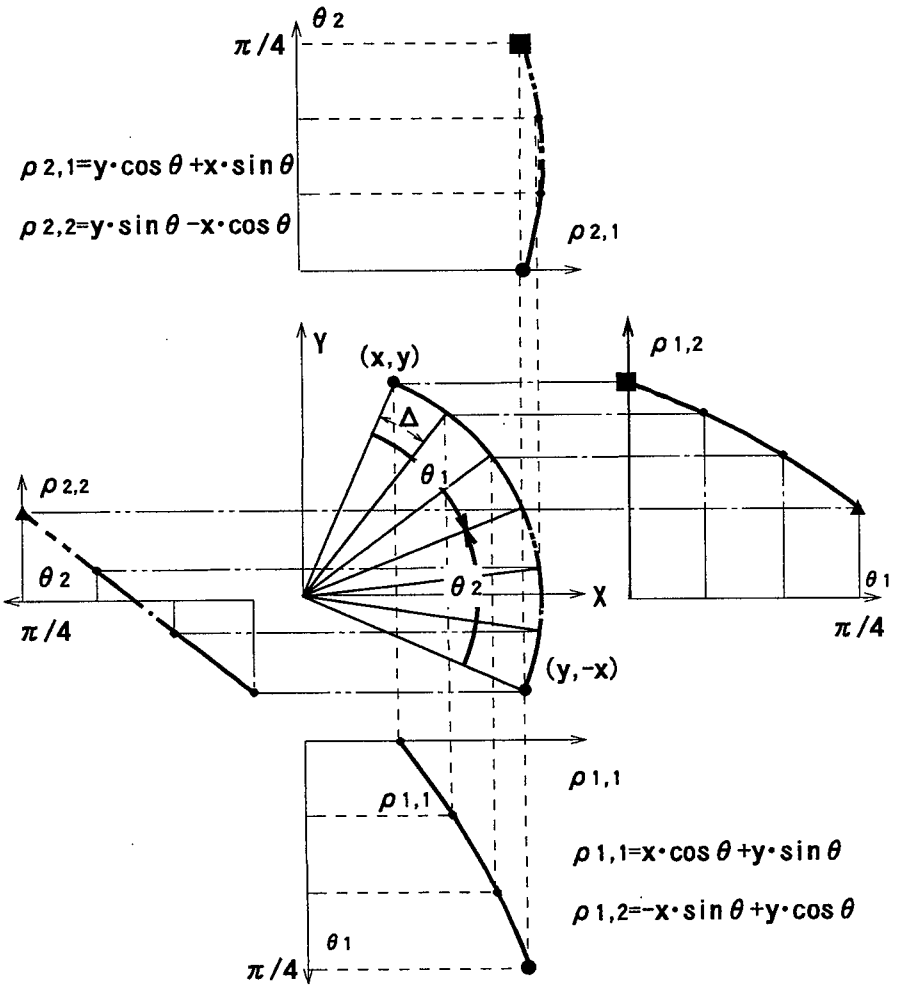


Fig.8.11　ρ Value calculation using quadruple parallel operation

$$\rho_{1,1,n+1} = \rho_{1,1,n} + \Delta \cdot \rho_{1,2,n} \quad \text{---(8.56)}$$

$$\rho_{1,2,n+1} = -\Delta \cdot \rho_{1,1,n} + \rho_{1,2,n} \quad \text{---(8.57)}$$

が得られる．但し初期値は$(\rho_{1,1,0},\ \rho_{1,2,0})^T=(x, y)^T$である．

である．また式(8.50),(8.51)より

$$\rho_{2,1,n+1} = \rho_{2,1,n} - \Delta \cdot \rho_{2,2,n} \quad \text{---(8.58)}$$

$$\rho_{2,2,n+1} = \Delta \cdot \rho_{2,1,n} + \rho_{2,2,n} \quad \text{---(8.59)}$$

が得られる．但し初期値は$(\rho_{2,1,0},\ \rho_{2,2,0})^T=(y, -x)^T$である．さらに(8.56),(8.57),(8.58),(8.59)に共通に$0 \leqq n < K_1 = 2^{m-2}\pi$である．ここで$\Delta = 2^{-m}$（$m$は正の整数）である．また$\theta$-$\rho$平面の寸法$K$，$K_1$，$L$は次のようにとる．$K=[2^m\pi]$，$K_1=[2^{m-2}\pi]$．但し[ ]は整数化を意味するガウス記号である．ちなみにm=7のとき，K=402，K_1=101となる．こ

こで図８．１に対応する図は図８．１２となる．このように式(8.56),(8.57),(8.58),(8.59)を組み合わせ，ハフ変換原式(8.1)を四重並列化して計算する近似式が得られた．これを実際にハードウエアに組み込むときの問題について次項で述べる．

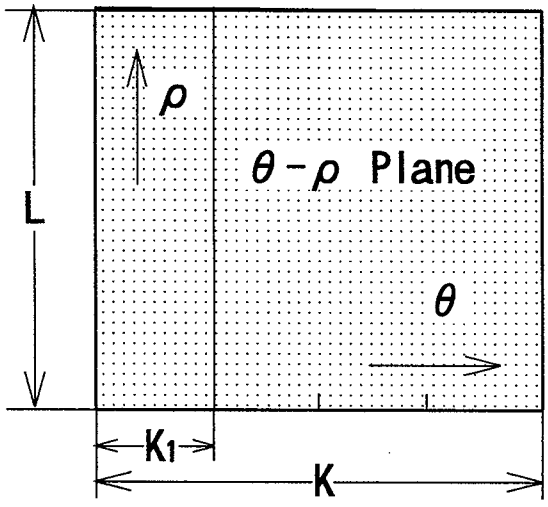


Fig.8.12 Dimension of K, K_1 ,L on $\theta - \rho$ plane

８．８．２　固定小数点演算と誤差の低減

前項で多重連立漸化式を導いた．ここではこの漸化式をハードウェアに組み込むためのρを表す数値形式の予備的な検討を行う．X,Y,θの軸分割数は全て402と仮定した．これはm=128としてK＝［$2^m \cdot \pi$］=402からθ軸分割数が決まるのでX,Yの軸分割数もこれに合わせたものである．この場合ρの最大値は$402\times\sqrt{2}$=568[Dot]となる．

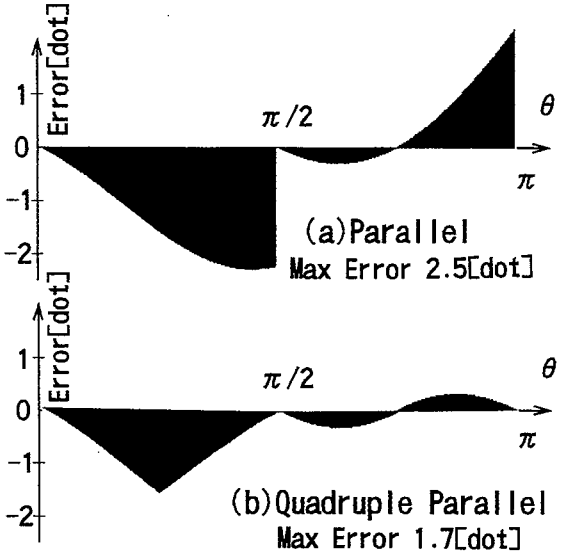


Fig.8.13 Error of two kinds of calculation method for original Hough transform expression (Floating point operation)

図８．１３は漸化式で使用するρを浮動小数点数で表現してHough変換原式(8.1)からの誤

差を表示したものである．与えたサンプリング点は誤差が最も大きくなるX,Y平面上の右上隅で$(x, y)=(402, 402)$である．図（ａ）は並列漸化式によるもの，図（ｂ）は前項で示した多重漸化式によるものである．初期値が$\pi/2$からも与えられたことにより$\pi/2$における誤差の段差がなくなると同時に全体として誤差が減少している．最大誤差は1.7[Dot]である．ハードウェアに組み込む際にはρ値を浮動小数点数で取り扱うと回路が複雑になると同時に演算速度の低下を招くのでここではρ値を符号及び小数部付き固定小数点数（以後単に固定小数点数と呼ぶ）として取り扱うことにする．整数部の桁数はρ値が最大567になるので符号桁まで合わせて11[bit]必要である．その場合小数部の桁数を何桁用意すればよいかを把握するためシミュレーションを行った．この結果を図８．１４に示す．図では小数部の桁数をsで表している．上から順にs=6～9の場合である．最大誤差はsが増えると漸減している．s=9でその値は1.7[Dot]となった．これ以上sを増加させても誤差の改善は見られなかった．従って小数部は９桁あれば必要十分である．

８．８．３　ハードウエア構成

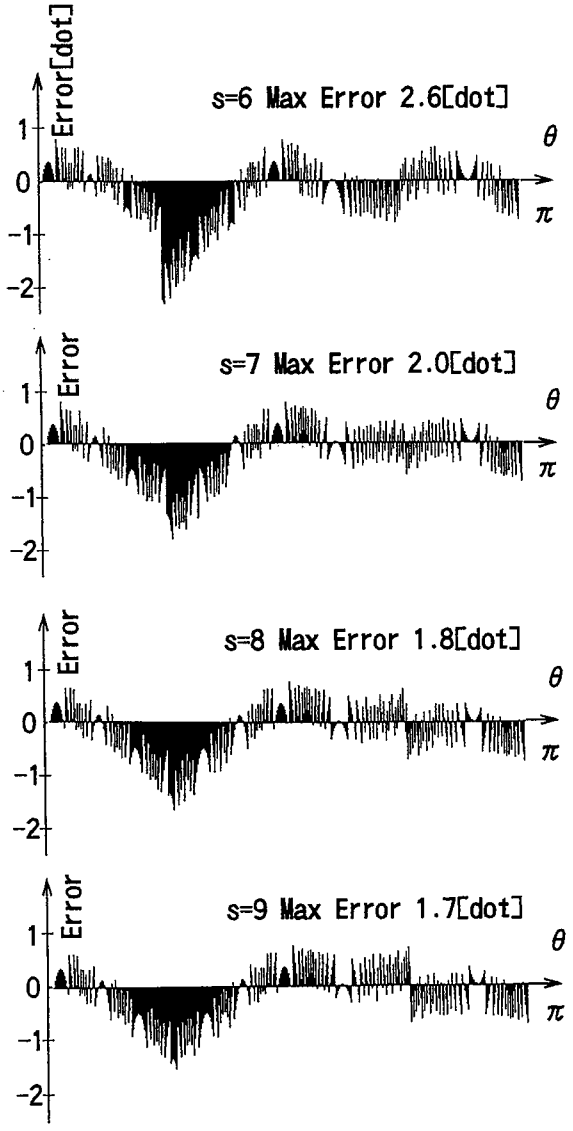


Fig.8.14 Error of quadruple parallel calculation for original Hough transform expression (Fixed Decimal point operation)

（１）試作回路のブロック図

本方式をハードウエアに組み込んだ様子を図８．１５に示す．主としてρ値発生部を中心に描いている．Latchはエッジトリガー形のレジスタである．TTL74S,TTL74ASシリーズのICで構成した．与えられた初期値$(x, y)^T$はそのまま時計方向回転行列演算部に入力される（図の上半分）．

一方，反時計方向回りの回転行列演算部にはxの２の補数が作られ$(y, -x)^T$が初期値として与えられる．（図の下半分）初期値設定後，スイッチが演算側に切り替えられクロックパルスの入力によって$\rho_{1,1}$，$\rho_{1,2}$，$\rho_{2,1}$，$\rho_{2,2}$が並列に発生されて行く．この部分へ供給されるクロックパルスはθ-ρメモリの度数累積加算部へも供給され，当該番地の内容が＋１される．図では省略されているが文献(47)に示すものと同じ回路が使用されている．但しρ値が２の補数形式で出力されるのでアドレス計算には多少の論理回路の追加が必要である．

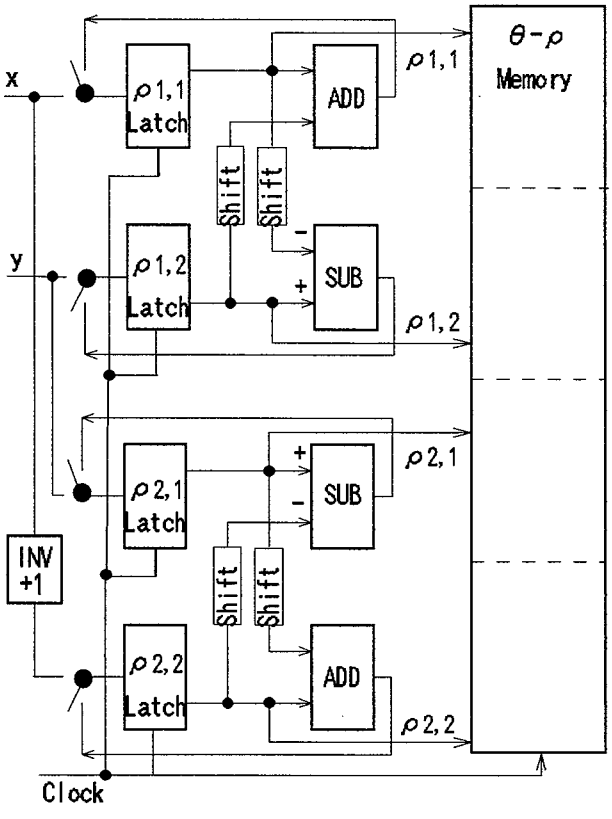


Fig8.15 Hardware block diagram of quadruple parallel calculation

この図で上半分と下半分は初期値の入れ替えを行えば全く同じ回路が使用可能なことがわかる．回路自体も同じような論理の繰り返しであるからPGA(Programable Gate Array)での構成が簡単でＩＣ化も容易である．

（２） ρ値のビット長

８．８．２項での検討からρ値のビット長は整数部と符号桁を含んで１１桁，小数部は９桁合計２０桁とした．このρが7bitのシフトを受けながら逐次演算が進むとすれば図８．

１６に示すように$\rho_{1,1}$の12～19ビットに$\rho_{1,2}$のMSBが足し込まれる．これは符号ビットである．また$\rho_{1,2}$の7～18ビットが$\rho_{1,1}$の0～11ビットに足し込まれる．結果として取り出されるρは離散的な整数部だけであるから$\rho_{1,1}$の0～8bitは切り捨てられる．

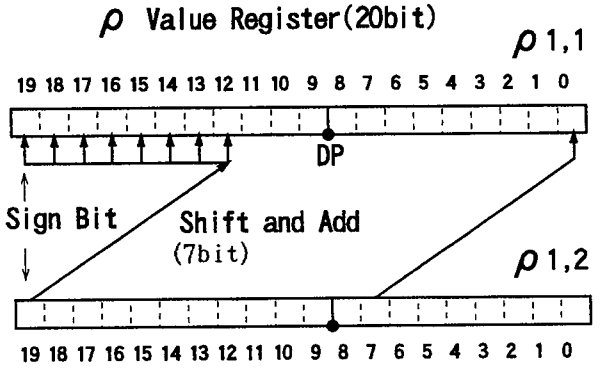


Fig.8.16 Bit shift for add and subtract operation

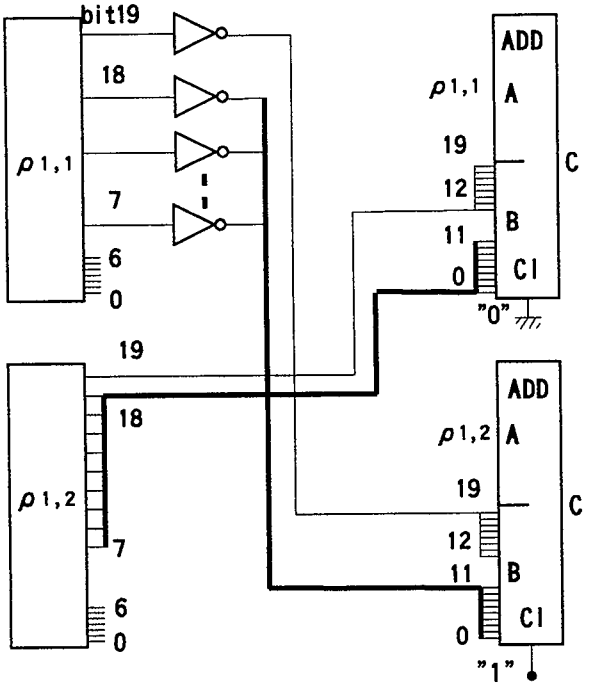


Fig.8.17 Detail chart of subtract operation part

（３） 減算部の具体的構成

本試作装置で取り扱うデータは２０桁の符号付き固定小数点データであり，その減算部（図８．１５，ＳＵＢ）は図８．１７のように構成されている．本図は図８．１５の左上部の詳細を示している．減算のための２の補数を作る部分を中心に描いている．図でビット線を表す数字はMSBより19～0と番号を振っている．アダー（ＡＤＤ）はＣ＝Ａ＋Ｂを計算する．最下部に下桁からのキャリーを入力する端子(CI)がある．シフト（Shift）は単なる線のつなぎ換えで済む．$\rho_{1,1}$は番号7～18ビットが反転されて１の補数となり下側のアダー（実質的には減算器）のＢ入力の0～11ビットに接続されている．特に$\rho_{1,1}$の

符号ビット，番号19は反転されてＢ入力の12～19ビットに接続されている．アセンブラプログラミングの算術シフトに相当する．そしてキャリー入力端子(CI)は"１"に固定されており"１"が加算され結果的に$\rho_{1,1}$の２の補数が完成し加算が行われる．補数作成がこのように特別な反転，＋１用のハードウエアを用意することなく自動的に行われる．$\rho_{1,2}$についてはこの値も負数になることがあるのでその符号ビット（番号19）は上側のアダーのＢ入力12～19ビットに並列に接続されている．

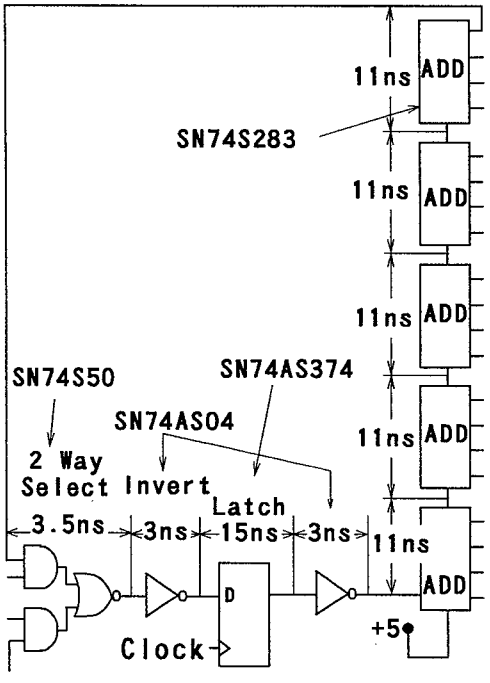


Fig.8.18 The longest loop of logic signal propagation

図が煩雑になるので省略したが上下２つのアダーのＡ入力にはそれぞれ$\rho_{1,1}$，$\rho_{1,2}$の全bit線がそのまま入る．

（４） ρ値伝搬のワーストケースと演算速度

図８．１８にρ値計算の信号の伝搬経路のワーストケースを示す．ワーストケースはρの最下位ビットからの桁上げが上位の桁のICの最上位桁まで伝搬するするときに発生する．同図はこのような状態を表している．2 Way select は初期値設定と逐次演算を切り替えるために設けられている．Latchは逐次演算をしながらρ値を更新していく．その出力が$\rho_{1,1}$，$\rho_{1,2}$，$\rho_{2,1}$，$\rho_{2,2}$である．Latchの次のインバータは2の補数を作るためのものである．アダー(ADD)はキャリールックアヘッド型の加算器である．図中に使用したＩＣの品名と伝搬遅延時間(TYP-中心値)を記入している．全体の遅延時間は計算上中心値で76.5[ns]である．

この中で最も大きな要素は加算器のキャリー伝搬時間，合計55[ns]である．市販のTTL ICでは4bit単位のものしか入手不可能なのでこのようになっている．8bitあるいは16bitの加算器が入手できればこの部分はかなりの速度改善がなされる．上記ワーストケースについて実験を行った結果を写真８．１に示す．測定点は図８．１８中のIC SN74AS374のCl

ock入力と（写真８．１(a),(b)の上段）とＤ入力（同下段）との間の時間遅れである．写真８．１(a)はClockが立ち上がってＤ入力が"1"->"0"に変化する場合，写真８．１(b)はClockが立ち上がってＤ入力が"0"->"1"に変化する場合を示している．遅れ量は後者の方が大きく約60[ns]である．これから計算するとクロックレートは約17[MHz]ということになる．

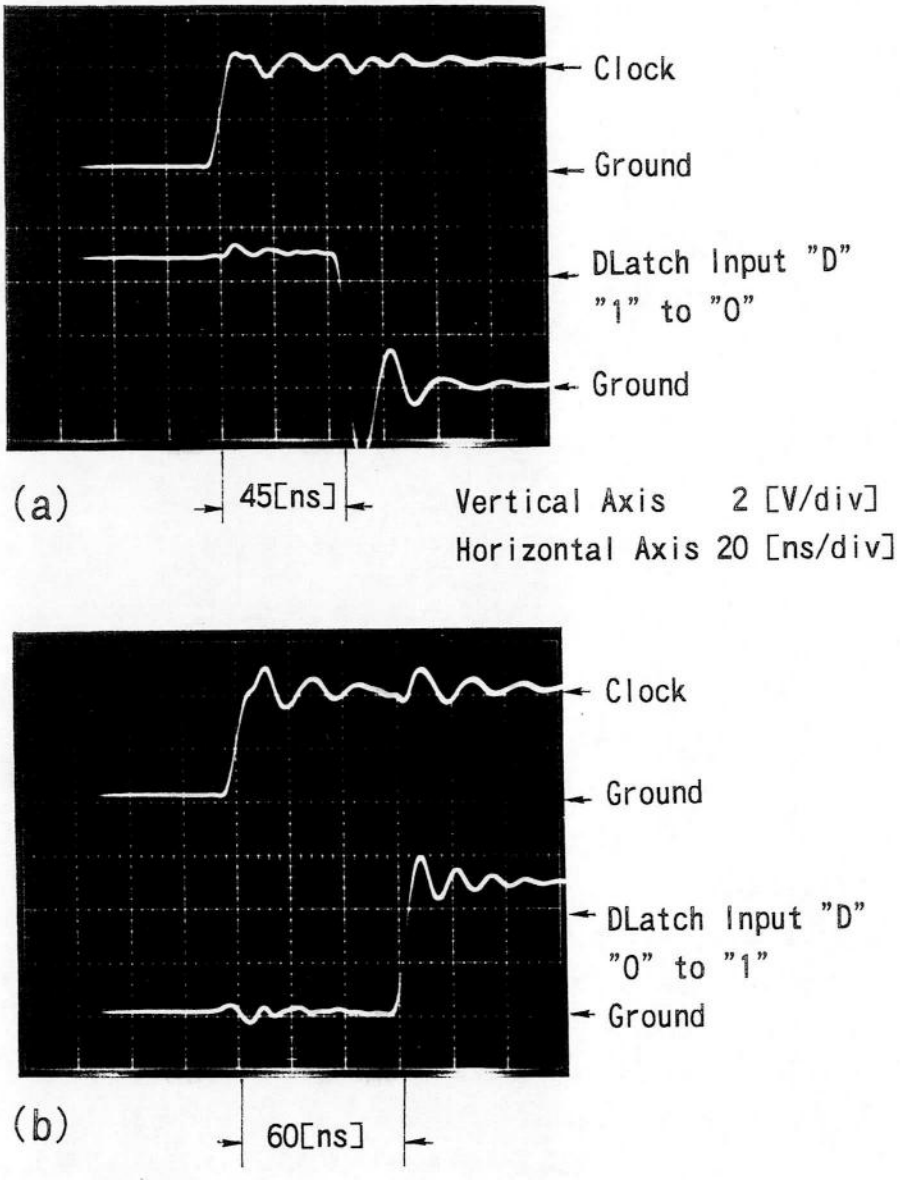


Photo. 8.1 Experimental result of propagation delay

８．８．４　考察

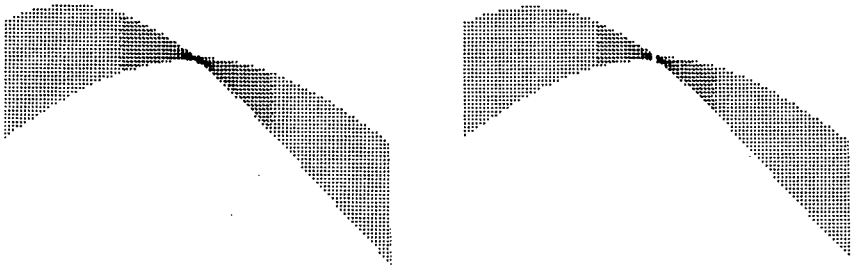
（１）　実行速度について

図８．１５の回路で，回路が安定に動作する最小クロック周期は上記60[ns]に余裕を持

って約70［ns］程度と推定される．θ-ρ平面の度数累積加算部は別の試作機で，一回の度数累積加算時間50[ns]を得ている[18]ので（度数累積部のハードウエアも四個用意する）ρ値発生部側の速度が若干遅い．両者はパイプライン的に実行可能なので全体の実行速度はρ値発生部で決まる．しかし４個のρ値が同時に生成されるので一つのρ値あたり18[ns]となる．結果的にはθ-ρ平面の一点あたり18[ns]となる．この速度は現時点でパソコンと外付けハードウエアの組み合わせとしては最高速と推定される．

（２） 精度について

文献(47)によれば本文で採用した並列式はFIHT2よりも誤差が大きく(X-Y)平面右端上部で大きさ2.5[dot]の誤差を発生していた．しかし，今回の多重並列式の場合，８．８．２項で述べたように初期値が２箇所（２組）から与えられるのでこの誤差は減少する．図８．１４より最大誤差はπ/4で生じその値は1.7[dot]である．単位[dot]はρ方向軸分割数の１単位である．この値はFIHT2による誤差とほぼ等しい．この程度の誤差は，原画像がサンプリングされる位置による誤差，２値化の誤差，微分によるエッジ抽出の際の誤差，度数ピーク点θ^*，ρ^*を求める際の誤差などの累積結果と適当にバランスする程度のものであり許容範囲にある．また図８．１４では図８．１３上段の$\theta=\pi/2$の位置における跳躍的な誤差の変化が無いことが大きな特長である．この跳躍的な誤差の変化があると，X軸に平行な図形縁辺点の座標を与えた場合，実際上ハフ曲線がπ/2の位置で接続する際段差を生じることになりピーク点位置が２つに分離することになる．図８．１９はこの例でX-Y 平面上でX軸に平行な直線をハフ変換演算しヒストグラムを形成したものの度数ピーク近傍を拡大して表示したものである．(a)は本節の４重並列方式によるものでスムースなハフプロットが得られているが[b]FIHT2を用いて描いた(b)ではピーク点近傍で線がとぎれている．



(a)Quadruple　　(b)FIHT2

Fig.8.19 Hough plot in the vicinity of π /2 on θ axis for horizontal line

８．８．５ まとめ

８．８節の内容を要約すると次のようになる．

（１）厳密漸化式を時計方向回りの回転行列と反時計方向回りの回転行列に適用し四重並列による高速演算を可能なことを示した．

(2)厳密漸化式から近似によって乗除算を含まない高速ρ値計算用の連立漸化式を導きρ値計算の4重並列実行を可能とした.
(3)論理信号の最大伝搬遅延経路に相当する回路を製作し処理速度を推定した.
(4)多重並列化の副次的効果として計算精度が向上した.
(5)それと同時に誤差の跳躍的変化が無くなりハフ曲線がスムースに接続されることが判った.

ホストコンピュータはこのハフ変換専用ハードウエアに初期値$(x, y)^T$を送出した後アイドルとなるのでこの間に8分割あるいは16分割用の初期値計算が可能であり,実時間性が厳しい用途にはこのような検討も必要となる.

8.9 結言

ここで,Hough変換を利用した図形パラメータ検出の過程を簡単に振り返る.その過程を第6章と同じく次の6段階に分けて考える.
①原画像が撮像装置によって捉えられ計算機のメモリイメージとなる.
②原画像を2値化する.
③2次元空間微分によってエッジを抽出する.
④Hough変換原式(8.1)によってθに対応するρ値を計算する.
⑤θ-ρヒストグラム平面への度数累積を行う.
⑥θ-ρヒストグラム平面の度数探索を行う.

この中で②,③については専用の画像処理用空間微分ICが一般に出回るようになり画像入力と同時に抽出されたサンプリング点座標が得られるようになった.また④のHough変換演算については本論文7章で挙げた表参照方式あるいは本章のいくつかの漸化式の提案により高速化の見通しが得られた.第7章で紹介した分割縮小化表参照方式の実験に使用したハードウエアではθ-ρ平面の1点あたり50[ns]という速度を得ている.このように②～⑤が本論文のような外部ハードウエアにより高速化されると相対的に問題になるのが⑥度数分布の探索に要する時間である.この部分はθ-ρ平面をθ方向,ρ方向に前後関係を見ながら度数ピークを見い出すのであるが手順的にハードウエア化するには荷が重い.この度数探索を補助する手段として次のような方法が考えられる.

θ-ρ平面の度数累積加算のためにその指定アドレス要素を読み出した時,一定以上の度数値を持つθ,ρアドレスをハードウエア上のバッファに蓄積しておき後でホストの計算機に読み込み度数探索の指針とする.

このことにより度数探索の範囲がある程度限定され探索時間が縮小されることになる.結局①～⑤をハードウエアで処理し⑥のソフトウエア処理に対してハードウエアが補助的に動作するという構成になる.この方式によればHough変換の画像入力から直線位置の特定までの手順を実時間で処理するシステムが十分実現可能である.

第９章　電荷蓄積素子による度数累積及び探索の高速化

９．１　緒言

Hough変換の際の度数累積メモリとして，２次元イメージセンサ(CCDイメージセンサ)をθ-ρヒストグラム平面メモリとして用いる，高速度数累積の新しい手法を考案した．その概要は第６章で触れた．本章では，その詳細な説明を行う．また試作ハードウエアの構成方法と評価結果を記す．最後に本手法を実施するための具体的方法と将来への展望を記す．原理は次のようである．

「計算機とは別に外付けハードウエアを用いる．ハードウエアはHough曲線発生部，ブラウン管オシロスコープ，２次元イメージセンサ，画像ＡＤ変換入力部からなる．X-Y平面上の変換候補点座標はHough曲線発生部に送られ，ブラウン管面上にHough曲線を描く，これを光学系によりセンサ面に投影する．曲線描画毎にセンサ中の単位画素に曲線の通過度数が電荷として累積される．最後にイメージセンサの内容をAD変換して計算機内に取り込む．この際，平行してヒストグラムの度数レベルを監視し，ヒストグラム探索の指標を得る．」

本手法は光学系を利用した２次元メモリの直接アドレッシングを用いてθ-ρヒストグラムの形成を行なっており，高速である．また描画光学系を複数個用いた並列処理への拡張が容易である．単一の回路を用いた試作回路ではθ-ρヒストグラム平面の１点当たり75[ns]の度数累積速度を得た．

Hough変換による図形パラメータ検出の過程を第６章と同じく次の６段階に分けて考える．

①原画像が撮像装置によって捉えられ計算機のメモリイメージとなる．
②原画像を２値化する．
③２次元空間微分によってエッジを抽出する．
④Hough変換原式(1.1)によってθに対応するρ値を計算する．
⑤θ-ρヒストグラム平面への度数累積を行う．
⑥θ-ρヒストグラム平面の度数探索を行う．

ここでは⑤θ-ρヒストグラム平面への度数累積を中心に考察する．度数累積は基本的にρ-θアドレスの特定要素の読み出し，度数累積加算，再書き込みの３段階が必要でハードウエア、ソフトウエア的にも直列処理となり,この考え方では高速化の余地は少ない．

そこで本章ではこの解決法として，２次元CCD(Charge Coupled Device)イメージセンサをθ-ρヒストグラム平面として用いる高速なヒストグラム形成法を提案する．それとともに，回路を実際に作成し，基本的な機能の確認と問題点の抽出を行った結果を述べる．最終的に，簡単な回路構成でθ-ρヒストグラム平面の１点あたり75[ns]で度数累積できることを確認した．⑥度数分布探索についても，θ-ρヒストグラム平面データの読み出し信号を監視して，探索範囲の指標が得られ，高速化が可能である．

試作した回路は簡素化のためHough曲線発生部に表参照方式を用いている．表はROM (Read Only Memory)で構成し，変換基本式を変形して表容量の縮小を計った．

提案した原理により，描画光学系を複数個用意すれば，θ-ρヒストグラム形成の過程

が並列に処理可能になり，Hough変換応用の実用的な障害であった処理速度の問題を解決する可能性を示している．

本手法は光学系を用いた２次元メモリの直接アドレッシングを利用しており，一種の光情報処理と考えることができる．本章の最後の部分ではシステム全固体化の具体的方法を提案している．これは将来実用化されるであろう立体化光電子回路の原型とも言える．

本章ではまず基本原理を述べ，その特長を示している．次いで試作した回路の構成と機能評価の結果を記している．さらに評価結果をもとに問題点及び適用限界を記すとともに，光情報処理，並列処理との関連において本法の発展的実用化のための展望も論じている．

図９．１は本章で提案する方式の概念図である．図形のエッジ抽出と外部ハードウエアで度数累積された結果の探索をホスト計算機内で行う．以後２次元CCDイメージセンサをCCDイメージセンサもしくはイメージセンサと略す．

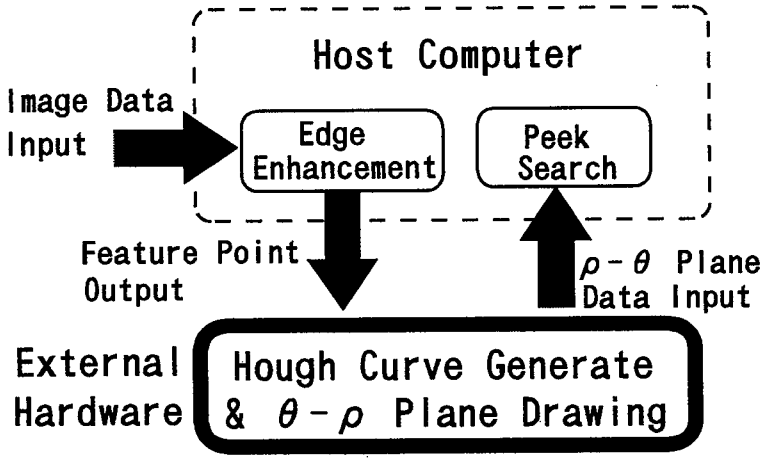


Fig.9.1 Fast Hough transfrom on external hardware

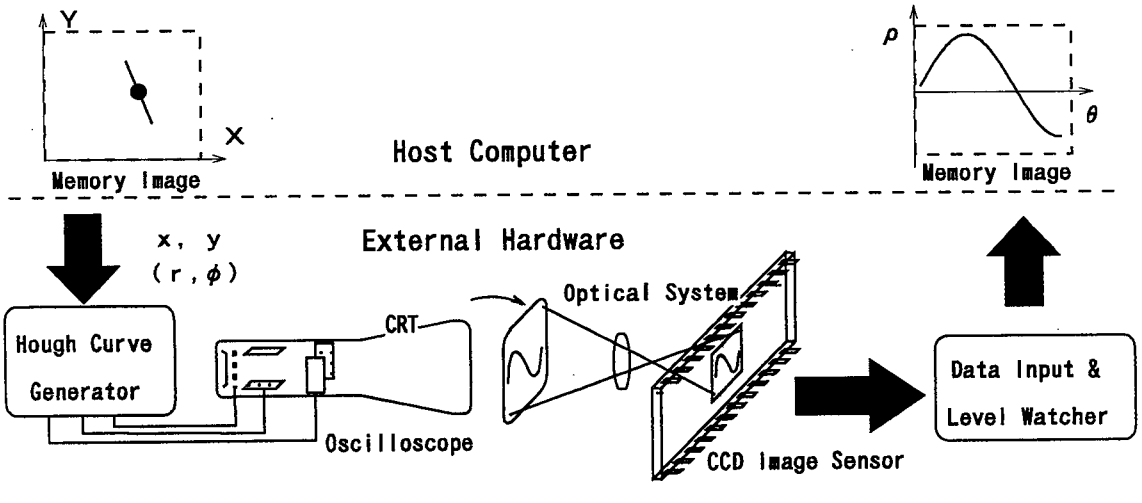


Fig.9.2 Principal of Hough transform employing Hough-curve generator and CCD area image sensor

９．２ 原理

図９．２に本手法によるHough変換処理の流れを示す．計算機内のサンプリング点データはいったん外部に接続されたハードウエアに移される．
ハードウエアはHough曲線の描画と度数累積を行い，結果は再び計算機内に戻され，度数探索処理がなされる．外部ハードウエアは次のような原理によるものである．

９．２．１　アドレッシングとヒストグラム形成の高速化

まず具体的な構成例を示す．図９．２のHough曲線発生部はX-Y平面上のサンプリング点座標(x, y)をもとに電気信号としてのHough曲線の変数θ，ρ値を発生する論理回路である(詳細は後に記す)．発生したθ，ρ信号をDA変換しこれをもとにしてHough曲線を２次元平面に描画する．（例えばθ，ρ値を図のようにそれぞれブラウン管オシロスコープの水平軸，垂直軸に加え，管面にHough曲線を描く）．このHough曲線像を光学系を通して２次元CCDイメージセンサに投影する．現在よく用いられているCCDイメージセンサの単位画素は図９．３(b)に示すようにフォトダイオードと積分コンデンサから成っている．図(c)はその等価回路で，Iは入射光量に比例した電流を発生する定電流源である．曲線軌跡が各画素を通過する毎に入射光量が電荷として蓄積される．これが計算機処理の場合のヒストグラム形成動作に相当する．

このように光学系でメモリアドレッシングを行い，電気アナログ的に積算記憶を行うのである．後節で試作回路について述べるが，これは問題点抽出のための実験であって，以上に述べた点が本章で主張する最も重要な点である．さて，すべてのサンプリング点について度数累積が終わればイメージセンサの内容はCCD転送チャンネルを通してシリアルに読みだされ，（図９．３(a)参照）AD変換されて計算機のメモリに読み込まれる．

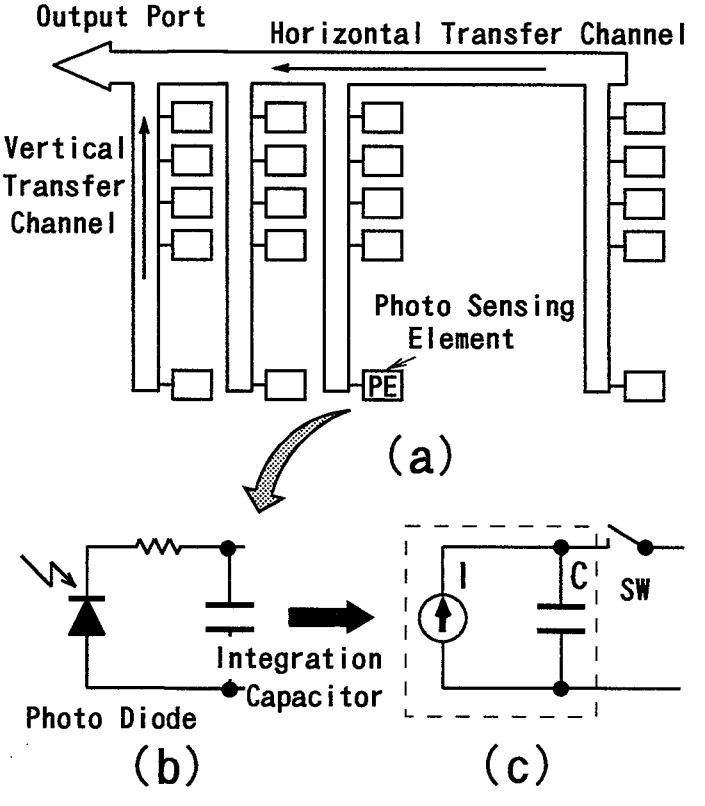


Fig.9.3　Internal structure of CCD area image sensor

図９．２の度数監視部(Level Watcher)は読みだしと並行して信号レベル（累積度数）を監視し，探索範囲に関する情報を得る回路である．この部分は応用の用途により構成が変化する．ここで得られた情報はFIFO(First In First Out)バッファに蓄えられ，θ-ρヒストグラム平面データの入力終了後読み込まれる．

９．２．２　Hough曲線の発生方法

高速化と回路の簡素化のため７．２節に示した表参照方式を用いた．コスト削減のため表サイズを小さくして，ROM１個で構成されるよう工夫をしている．この方法は配線量が少なく，簡単であったからである．表にはＲＯＭ（Read Only Memory)を用いた．このため後述のように度数累積速度がＲＯＭのサイクルタイム（規格値=120［ns］）に支配され，処理速度が原理的に考えられる速度よりも遅くなっている．ＳＲＡＭを用いれば10[ns]前後のサイクルタイムの物が市販されているので更なる高速化が可能である．

９．３　試作ハードウエアによる原理の検証

原理の確認と問題点の抽出を目的としてハードウエアを製作した．図９．４にHough曲線発生回路のブロック図を示す．サンプリング点座標(x, y)は(r, ϕ)に変換されてホスト計算機から与えられる．ホスト計算機は(r, ϕ)を送出しハードウエアによる度数累積が始まった後，次の(r, ϕ)の計算を行う．初期位相ϕはラッチカウンタにプリセットされ，ωはこの値から１ずつ増加する．θのカウンタは最初０にクリアされ１ずつ増加する．ROMはrとωをもとにρを送出する．その後θ，ρの値はDA変換器を経てオシロスコープのそれぞれX,Y軸に加えられる．描画していないときスポットを消すため，オシロスコープのZ軸（輝度信号入力端子）制御信号発生回路を設けている．θ-ρヒストグラム平面は第７章に述べたものと同じ方法で原点移動と縮小変換を行った．

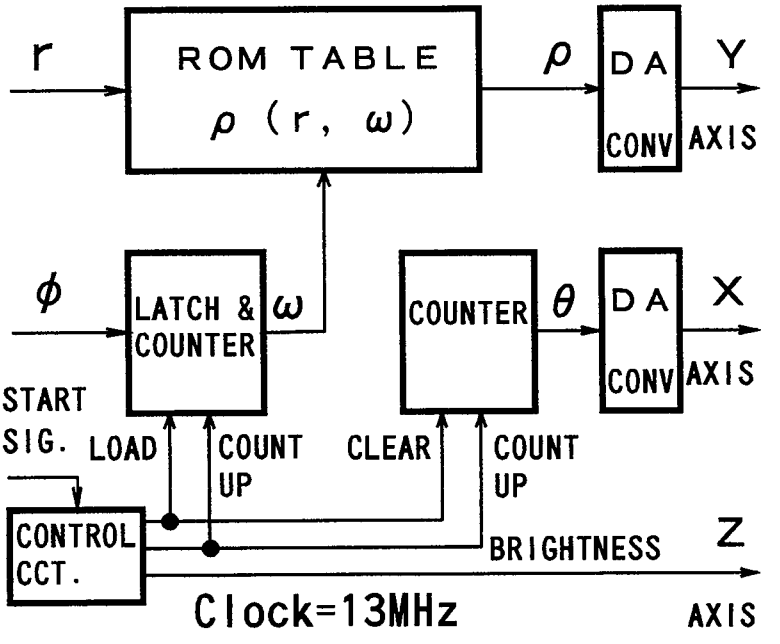


Fig.9.4 Block diagram for Hough curve generator.

軸分割数は$J=K=L=M=256$(8bit)である．r，ϕのビット長は8である．DA変換分解能は8bitである．制御部を除いて各ブロックがIC１個に相当する．θ-ρヒストグラム平面のθ信号の代わりにオシロスコープの時間軸を利用してもよいが，精度向上のため系のビット長を増加したときに，水平掃引の時間軸直線性不良の影響で誤差が出る可能性があるので，別回路を設けた．オシロスコープの表示面とイメージセンサ素子面との不整合は誤差発生の大きな要因となる．オシロスコープのゼロ点は動き易く且つ湾曲する可能性があるので校正は重要である．この校正は次のようにして行う．

[1]x,yの最大値を与えHough曲線を発生させ，θ信号を0として表示面に縦軸(ρ軸)を光らせる．

[2]計算機に読み込みその位置をρ軸とする．

[3]x,y共に0を与えHough曲線を発生させ表示面に横線（θ軸）を光らせる．

[4]計算機に読み込み，その位置をθ軸とする．

回路のクロックは13[MHz]である．使用したICの総数は9個であり非常に少ない(パソコン本体の入出力ボード登載ICを除く)．イメージセンサとしては画像蓄積と読出しを外部コントロール可能な市販のCCD工業用モノクロカメラを利用した．

９．４　性能評価

（１）　度数累積速度と実験結果

θ-ρヒストグラム平面の１点あたり75[ns]で度数累積できた．曲線１本（θ軸分割数M=256)にすれば20[μs]である．　周波数に直すと50[kHz]であり，オシロスコープの帯域幅は広い必要はない．但しZ軸応答は高速でなければならない．光学系に使用したレンズの口径は21mmであるがCCDセンサの感度はこの描画速度に対して十分であった．

図９．５は計算機内で発生させた大きさ256×256の正方形上の点を試作ハードウエアに与えてHough曲線を描画させ，ＣＣＤカメラを通して再び計算機内に取り込んだメモリイメージをプリントしたものである．濃度階調は16レベルで軸湾曲に対する補正を行っている．試作ハードウエアのθ軸分割数は256であるが図はプリントサイズの関係で128に縮小している．この程度の分割数では光学形の歪，その他の非直線性の影響は目視確認できない．上図の辺A,B,C,Dに対応するピーク点も明瞭に出ている．

（２）　θ軸について

θ軸をDA変換を用いて発生させたが，オシロスコープの時間軸を利用してもこの程度の分割数(256)では，目視上差はなかった．

（３）　誤差の発生と適用限界

ROM のビット数（分解能）の問題は理論的に推定できる．その他の誤差の発生要因はオシロスコープのスポット径がある程度以上は絞れないこと，偏向の非直線性，光学系の歪，DA/AD変換の非直線性などである．

これらを考慮すると通常のオシロスコープと組み合わせたこの方法の場合，軸分割数L,Mは512が限界と考えられる．

イメージセンサの0レベル付近のノイズにより度数1との判別が難しい．従ってHough曲線の包絡線の幅を利用するような応用[1]には適しない．

（４） 表示面と受光面の整合及び感度非直線性等

X-Y，θ-ρ軸の位置合わせは大きな問題である．管面のうねり，X,Y偏向系の非直線性，カメラを含めた光学系の歪などが影響する．計算機に読み込む際，θ-ρメモリの外側にガードエリアを設けて,ソフト的に整合を計るべきである．

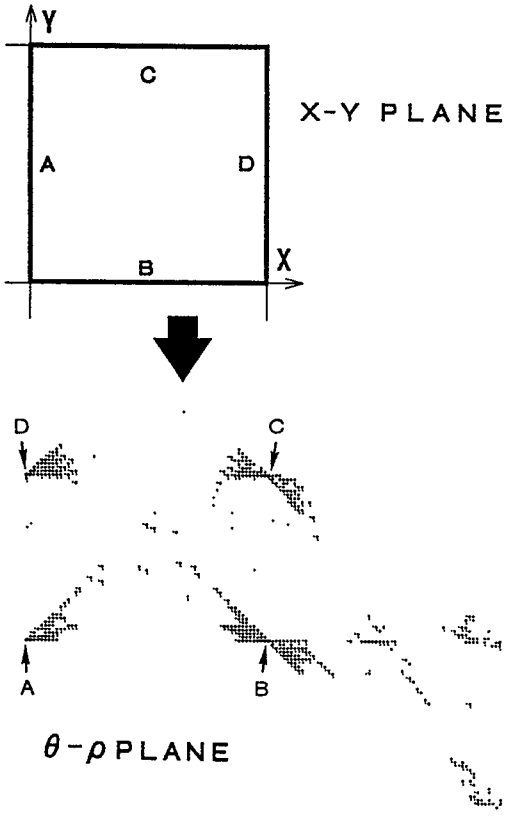


Fig.9.5 Rectangular edge and obtained Histogram image

またイメージセンサの面内感度不均一性，画素値の走査回数に対する非線形性などによるピーク点位置の不安定性の問題もある．前者については均一輝度のテストパターンを用いて，あらかじめ不均一性の度合いに関する情報を得ておき，ピーク検出の際に補正を加えることで解決できる．後者については，ピーク検出操作が一種の微分操作であることを考えると，隣接する画素間で極端な線形度の違いがなければ，誤ってピーク検出する可能性は少ないと考えられる．スポット径の問題について言えば，ブラウン管上の輝点は輪郭のはっきりした円ではなく，中心部が明るく周辺に行くに従って同心円的に暗くなっている．このためＣＣＤセンサ各画素(575H×485V)の物理的な寸法，配列を考慮すると，輝点の中心部は位置精度に応じた正確さで目的の画素を走査していくことになり，試作装置での実験結果（図９．５）ではスポット径が有限サイズであることによる影響は明確には表れていない．

（５） 輝度の制御

Z軸制御信号は元々オシロスコープの帰線信号を消去するために設けたのであるが，サンプリング点の確らしさに応じて輝度レベルを制御し，１回の累積量を変化させることも

可能である．

９．５　ＬＣＤを用いて全固体化する方法

本手法は立体空間を利用したメモリアドレッシングを用いており，光情報処理の一種と考えられる．この方法は高速処理を行う上でユニークであるが物理寸法的に大きくなる．また９．４節に挙げた問題で高精度化には限界がある．次にこれらの問題について検討を加える．

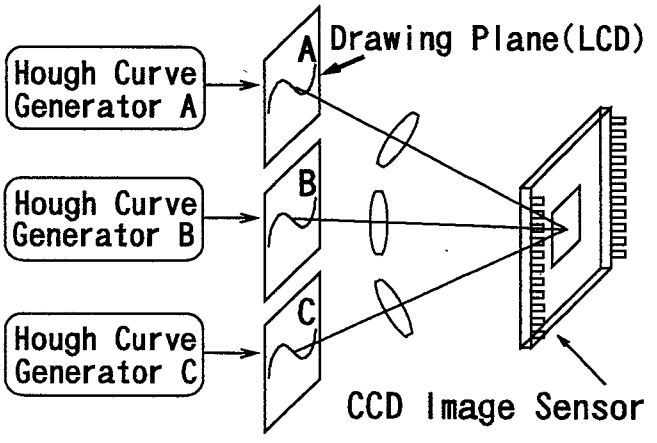


Fig. 9.6 Parallel processing employing multi-drawing plane

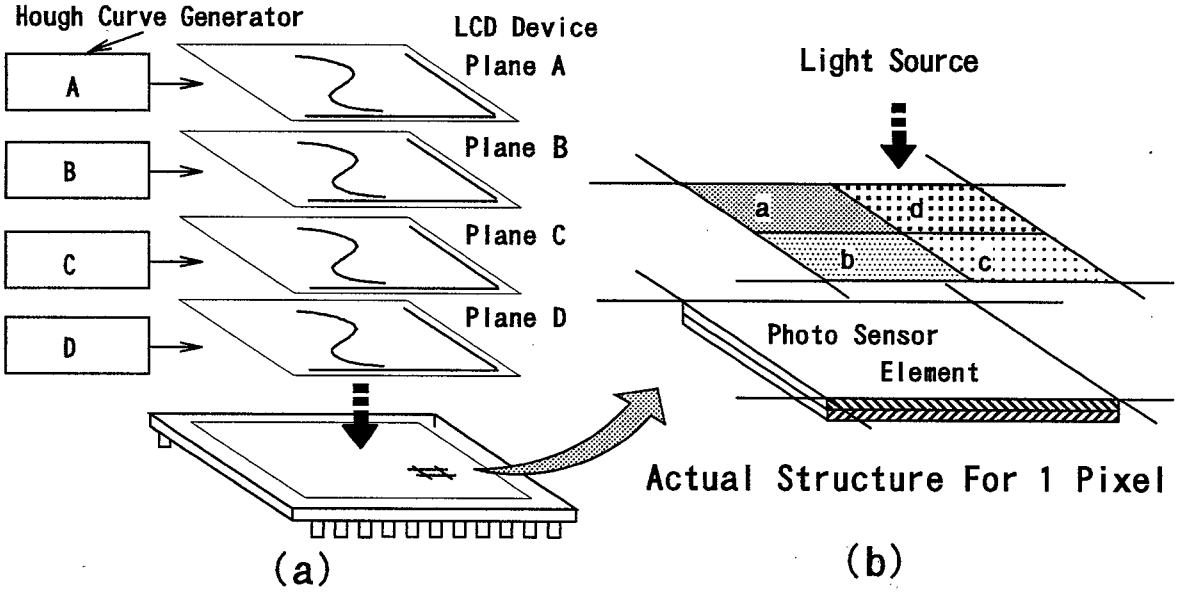


Fig.. 9.7 Parallel processing employing LCD multi-drawing plane and its physical structure for unit cell

本手法の本質は次の２点である．

[1]光学系を用いることによる２次元メモリの高速アドレッシング．

[2]２次元CCDイメージセンサのアナログ積算素子としての利用．

実用化のためにはレンズ等を用いた光学系をなくし小型化，固体化の必要がある．オシロスコープ管面をLCD(Liquid Crystal Device)に置き換え，更に並列化を計ると図９．６のような度数累積システムが考えられる．プレーン A,B,CはLCDを用いた度数累積面である．この度数累積面上に与えられたサンプリング点複数個について，並列に描画を行い，この像をイメージセンサ面に投影する．

図９．７は前図からレンズ系を取り除き全固体化を計ったものである．図の左半分は４層の描画面の像がイメージセンサ面に投影される様子を概念的に示している．この方法で下部の描画面がより上部の描画面からの映像の通過を妨害することを防ぐために，同一平面内でプレーンが分割されて，各映像情報が並列に加算されるように構成する．右図はこの構成の１画素分を取り出して示している．a,b,c,dはプレーンA,B,C,Dの１画素相当分を示す．このシステムを実現しようとすると，現状ではLCDの応答速度，高密度集積化の問題が残されている．

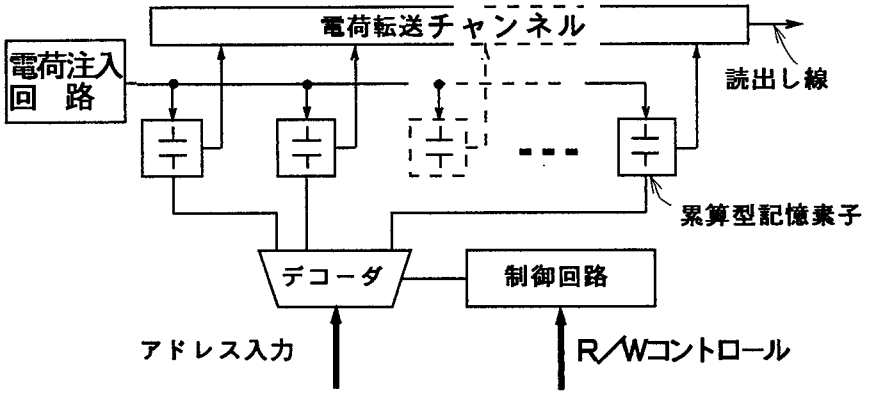


Fig. 9.8 Block diagram of special LSI

LCDはLED(Light Emitting Diode)やレーザースキャナ等の高速デバイスに置き換えることも考えられる．集積化については，構成すべき回路が同一回路の繰り返しなので，ウェーハサイズの製造上の問題が解決すれば大きな問題ではない．

図９．８は物理的に大きくなった試作装置をＬＳＩ化した場合の概念図である．光学系は取り除かれ，アドレス線の直接的な指示により特定のアドレスに直接電荷を注入する方法である(特許1)．Hough変換が多くの場所で使用されるようになった場合にはこのような専用ＩＣが必要になると思われる

方式の理論的な検証が目的であるから，簡素化のためHough曲線発生部に表参照方式を用いた．処理速度の制限要因は表としてのROMのアクセス速度である．高価ではあるがより高速なSRAM(Static Random Access Memory)を用いればθ-ρヒストグラム平面の１点あたり30[ns]以下が実現できると考えられる．

９．６　結言

現時点では，アクセス速度が100[ns](規格値)以下のROMは市販されていない．これは製造技術上の問題であるから，将来的にはSRAM程度まで高速化されるであろう．そうなればCCDイメージセンサの感度には余裕があるので，９．５．２項に述べた速度30[ns]はROMを用いて実現可能である．本章冒頭に述べた⑥度数分布探索については，本章の方法によれ

ば度数監視部で探索指標が得られるので速度向上が期待できる．

本章で述べた試作回路は直ちに実用の装置として供する事は困難な点が多い．しかし技術革新が急速に進みつつある光情報処理，立体電子回路の技術により，９．５節で述べた本手法の個体化が実現できれば，Hough変換の実時間処理への応用は大きく拡大する．これをツールとして利用するアプリケーション分野について研究が必要になる．

本章の内容をまとめると次のようである．

(1) 2次元CCDイメージセンサを積算形アナログメモリとしてとらえ，Hough変換のθ-ρヒストグラム平面メモリとして利用する考え方を示した．

(2) 2次元メモリの高速直接アドレッシングを光学的な手法を利用して実現した．

(3)電子回路で電気信号としてのHough曲線を発生させ，オシロスコープ，光学系,CCDイメージセンサを組み合わせた高速θ-ρヒストグラム形成法を提案した．実際に回路を構成し機能確認，問題点の抽出を行った．

(4)参照表のサイズは汎用中容量のROM１個でよかった．度数累積速度はθ-ρヒストグラム平面の１点あたり75[ns]であった．

(5)θ-ρヒストグラムのデータを計算機に再入力する際，θ-ρヒストグラム平面探索の補助情報が得られ探索の高速化が計れることを示した．

(6)本手法を高精度化，実用化するための方法を考察し将来への展望を示した．

１０章　Hough変換の応用例

１０．１　緒言

本章ではHough変換の応用例を３つ挙げる．第１１章で実時間Hough変換を実用に供したシステムについて述べるので本章の例は特に実時間性を意図したものではない．但し若干の速度に関する考察を加える．一つは溶接部材の開先形状をレーザービーム光スライス法により求めたものである（１０．２節）．この例については８章で述べた連立漸化式を用いてHough変換をソフトウエア上で行ったもので，同章で述べたいくつかの漸化式について比較検討を行っている．次はHough変換により抽出した直線を基準点として利用する，物体の３次元形状計測の基礎実験の例である（１０．３節）．ここでは奥行き計測の簡単な原理と測定結果及び測定精度について述べている．最後に道路上センターラインの検出の例を挙げる（１０．４節）．この実験は将来的に自動車の自動操舵につながる基礎実験であって，道路環境が整備されればそのような自動運転も可能となる．但しこの場合は実時間のHough変換装置が必須となる．

１０．２　溶接部開先形状の測定

写真１０．１は溶接部材開先部に線状の半導体レーザ光を照射して，光スライス法により得られた開先形状の像である(画素数512×480)．このデータはフロッピーディスクに取り込んだディジタルデータとして入手したものである．このような像から開先角度，ルートギャップなどが測定できるが，この目的で写真１０．１の明部データを対象にしてHough変換を施し，線分要素を認識した．使用した実験システムのハードウエア構成は第５章５．３．１項、図５．３に示したものとほぼ同じで，処理を行うソフトウェアが異なるだけであるからここでは掲載を省略する．但しホストとなるパソコンが異なる．写真１０．２はHough変換による直線検出の結果を示すもので図中の（Ａ）の直線についてその位置はFIHT2式，並列漸化式でそれぞれθ=22[°]，ρ=262[dot]，高精度漸化式でθ=22[°]，ρ=263[dot]と求められた．

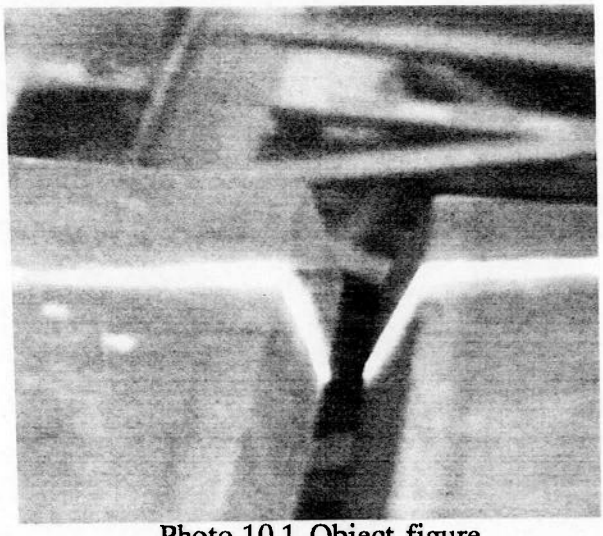


Photo.10.1 Object figure

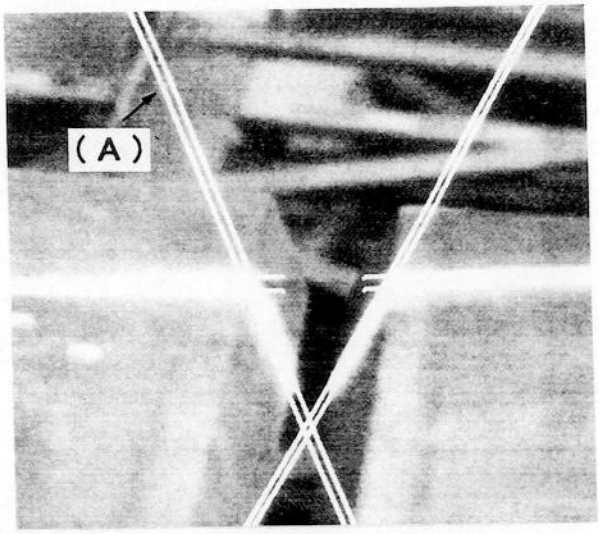


Photo.10.2 Obtained lines by Hough transform

この程度の原画像の品質ではどちらが正しいかは判断できず有意差は認められない．換言すればいずれの方法でも適用可能だと言うことである．もう一つの問題，すなわち処理時間についてはソフトウエア処理の場合，原画の２値化に51[ms]，エッジ抽出に275[ms](Robertのオペレータ[51]採用)，θ-ρ平面への累積加算に160[ms]，θ-ρ平面の度数探索に360[ms]要している．問題のHough変換演算にはFIHT2式，並列漸化式，高精度漸化式の順に950[ms]，950[ms]，1200[ms]という結果が出ている．X-Y平面上のサンプリング点の数は1123であった．環境はPC-9801FA i80486 66[MHz]，MS-DOS V3.3，言語はTurbo C V2.1である．高精度漸化式が予想以上に速いのはCPUが浮動小数点コプロセッサを内蔵しているからであると思われる．一連の処理をアセンブラで記述すれば，特にFIHT2式，並列漸化式によるHough変換演算などにおいて，より高速化が期待できる．さらに並列漸化式では前章，前前章で述べたようなρ値計算のハードウエア化に，市販のエッジ抽出用前処理モジュールを用いれば全体として処理時間を1000[ms]以下にすることは容易であると推定できる．この例では処理を外部ハードウエアを用いずに行ったがかなりの実時間性がある．

１０．３　３次元形状計測への応用

１０．３．１　立体視とステレオ写真

人間が物を立体的に見ることができるのは，左右の目が少し違った角度から物を捉えているからである．

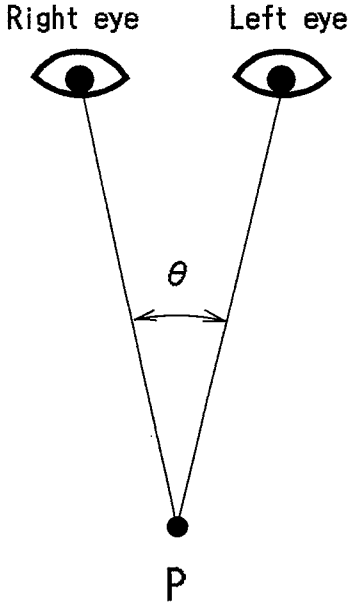


Fig.10.1 Observation of the point of 3−dimensional space

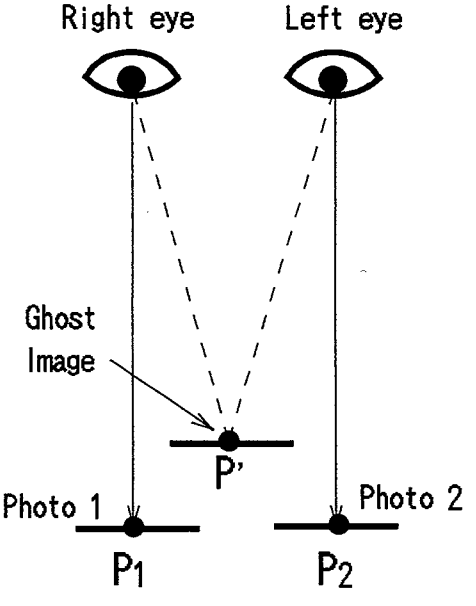


Fig10.2 Schematic of stereo−graphic observation

この場合の立体感の強弱は図１０．１に示すように，物と左右の目で作られる２等辺三角形の頂角θの大きさで決められる．例えば遠くの物が平面的に見えるのは角度θが非常に小さく左右の目で見た像の間にほとんど差がないためである．この原理を応用し，立体視を行うための２枚の写真をステレオ写真といい，２枚の写真を用いて立体視を行ってい

る様子を図１０．２に示す．一般の観察には２枚の写真を左右に並べ立体鏡を用いて行う．左右の目で各々対象とする同一な部位だけに着目し，左右の写真を動かすと２重に見えていた対象部分があたかも立体的に感じられるのは前の理由からであり，立体虚像を見ていることになる．ステレオ写真は基本的には同一視野で同一倍率の対象物に対しての視点からある角度傾斜させて撮影したものでその撮り方として２種類が考えられる．一つは固定された点から被写体を回転させてとる方法（図１０．３（ａ））で顕微鏡写真のようなミクロな対象に使用される．今一つは被写体から等距離の点にある異なった２点で写真を撮る方法（図１０．３（ｂ））で空中写真のようなマクロな対象に適用されている．両手法を比較してみると，対象物が動かせない場合には後者の方法しかとることができない．逆に顕微鏡写真のような場合には前者の方法しかとることができない．ここでは後者の，撮影位置を動かしてステレオ写真を撮影する方法を用いることにした．

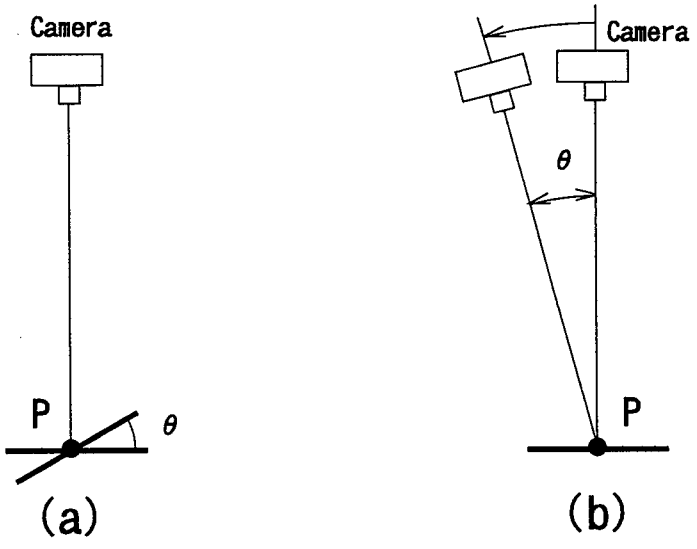


Fig.10.3 Schematic of stereo-graphing

１０．３．２　３次元形状計測の概要

前節で作成したステレオ写真から写真中の任意の点の深さや高さを求める方法を述べる．

図１０．４は基準となる平面から奥行きhの点Pがステレオ写真でどのように投影され，カメラに取り込まれるかを示したものである．ここで基準となる平面Sは紙面に垂直で最初にＡの方向から撮影するものとする．ここでカメラと目標点の間隔は奥行きに対して十分大きく，写真乾板上には点Pの平行光線による投影が得られるものとする．このとき目標点Pはカメラを回転するときの軸(Pivotで表す．)からP_1の長さに投影される．次にカメラの位置をl_2だけ上方向に移動してＢ方向から撮影する．カメラの光軸はPivotに合わせておく．ここでカメラを回転移動ではなく直線移動したのは後に示す実施例に合わせたためである．さてこの位置からの点Pの射影はPivotの位置からP_2に投影される．

ここで　$P_2=P_1 \cdot \cos\theta + h \cdot \sin\theta$　従って

$$h=\frac{P_2}{\sin\theta}-\frac{P_1}{\tan\theta} \qquad \text{---(10.1)}$$

と表せる．Pivotとしての平面内にある直線（紙面に垂直）はHough変換によって検出できる．また点Pについてはパターンマッチングにより長さP_1,P_2を単位を[Pixel]としてその長さを測定できる．またl_1,l_2は撮影時に測定すればよい．θは$\theta = tan^{-1}(l_2/l_1)$である．．従って(10.1)式より$h$が単位を[Pixel]として求まる．但しこのときのhは図１０．４でカメラを垂直移動したためにPivotからの距離が$l_1/cos\theta$に伸びた状態で撮影されることになる．被写体からの距離が伸びると反比例して像は小さくなるのでP_2にはこの補正が必要になる．この補正は前述の平行光線による投影の仮定と矛盾するが被写体とカメラ間の距離が有限で且つＢ方向からの距離がl_1に対して無視できないので計算に取り入れた．

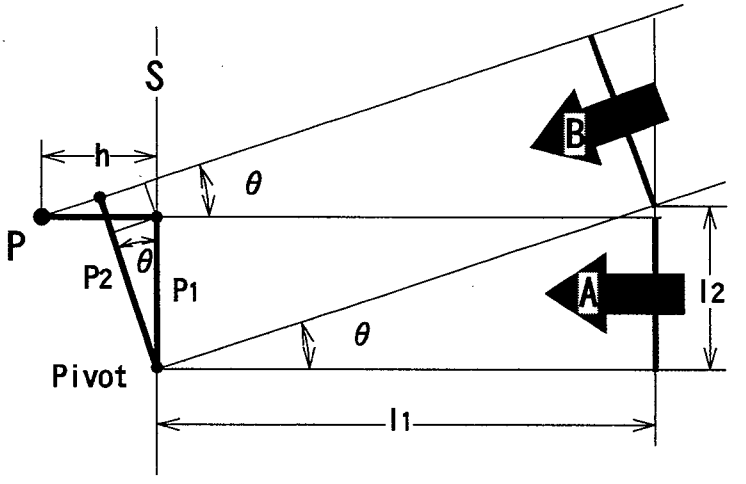


Fig.10.4 Schematic of stereo-graph

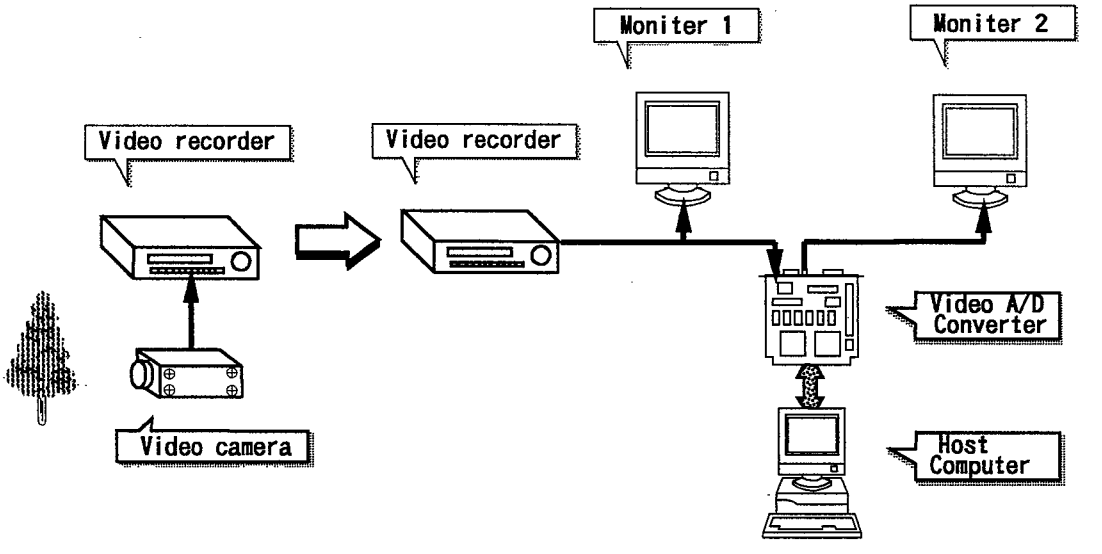


Fig.10.5 System Configuration

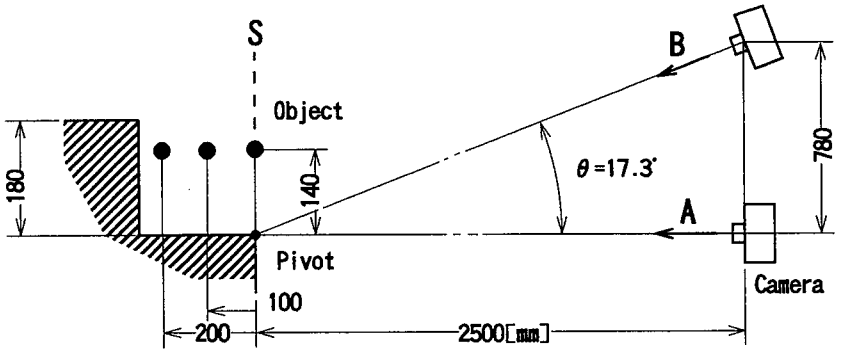


Fig.10.6 Setting of examination

更に単位を［Pixel］として求めたhを実寸に換算するには一度だけP_1或いは適当なS平面内の図形の実寸を測定しておき，係数K［mm/Pixel］を求めておけばhの実際の長さh_rは

$$h_r = K \cdot \left(\frac{P_2}{\sin\theta \cdot \cos\theta} - \frac{P_1}{\tan\theta} \right) \qquad \text{---(10.2)}$$

となり注目している点Pの平面Sからの奥行きが単位を［mm］として得られる．

１０．３．３　実験システムの構成

図１０．５に本実験に用いたシステムの構成を記す．原画像はビデオカメラにより撮像されビデオレコーダによりビデオテープに記録される（ＶＨＳ形式）．

これを本体システムのビデオレコーダで再生しイメージＡ／Ｄ変換ボードに入力する．このボードの解像度は512×504[Pixel]である．再生された画像はモニタ１で監視できる．Ａ／Ｄ変換ボードはディジタル信号に変換された画像データを再度Ｄ／Ａ変換して出力できるようになっている．ボード上には256[kb]のRAMが搭載されておりこれはホストコンピュータのメインメモリにバンク切り替えで接続されている．１画素当たり１バイトが割り当てられておりグレースケールは256階調である．このボードは２つの動作モードを持っておりいずれかを選択することが出来る．一つはホストコンピュータからメインメモリの一部としてアクセスできる状態、今一つは先に述べたようにボード上のディジタルデータをＤ／Ａ変換してモニタ２に出力する状態である．ホストコンピュータはPC9801 RX i80286 12[MHz]である．

前節では原画像データをフロッピーディスクに記録されたものを用いた．本節でのシステムとは別であって前節のそれに比して処理速度は1/10程度である．

１０．３．４　実験結果

実験は写真１０．３のようなビルの階段付近を使用して行った．写真には検出された主な直線を重畳している．このような場所を選んだのは階段部分が多くの直線部で構成されており，Hough変換の直線検出の性能を試験するのに適当な場所だからである．図１０．６は立体視の試験を行う状況の説明図で斜線部は階段の断面を示している．カメラはPivotを含む平面Sから2500[mm]の距離にセットした．高さは平面SのPivotに垂直な位置である．目標物は高さ140[mm]の位置に直径10[mm]の球を取り付けている．写真１０．４は目標物

がないとき，図１０．４のＡ方向からの写真である．次に目標物をPivotから0[mm]，100[mm],200[mm]の位置に置き撮影した写真がそれぞれ，写真１０．５，写真１０．６，写真１０．７である．次にカメラを以前の位置から垂直方向780[mm]上方に移動しカメラの光軸をPivotに合わせ（１０．４のＢ方向）前と同様に目標物を移動させながら写真１０．８，写真１０．９，写真１０．１０を撮影した．写真１０．５～１０．１０にはHough変換により求めた直線を重畳している．

こうして得られたコンピュータ画像から(10.1),(10.2)式を用いてS平面からの奥行きh_rを計測した結果とスケールによる実測値，誤差を表１０．１に示す．なお画素レベルでのP_1,P_2の長さと実寸の換算係数KはＡ方向から見たとき（写真１０．４）の階段の一段当たりの高さ180[mm]に対する画素を単位とする長さを用いて求めた．K=1.224[mm/Pixel]であった．また$\cos\theta$は$\cos\theta$=0.955であった．

１０．３．５　考察

表１０．１で誤差の欄は測定値　(h_r／真値－１）×100[%]で示している．奥行き0[mm]のとき誤差が大きい．これは階段の角の部分を危険防止のため樹脂のカバーで丸めてあり高さと奥行きを正確に140[mm],0[mm]に設定することができなかったためと考えられる．

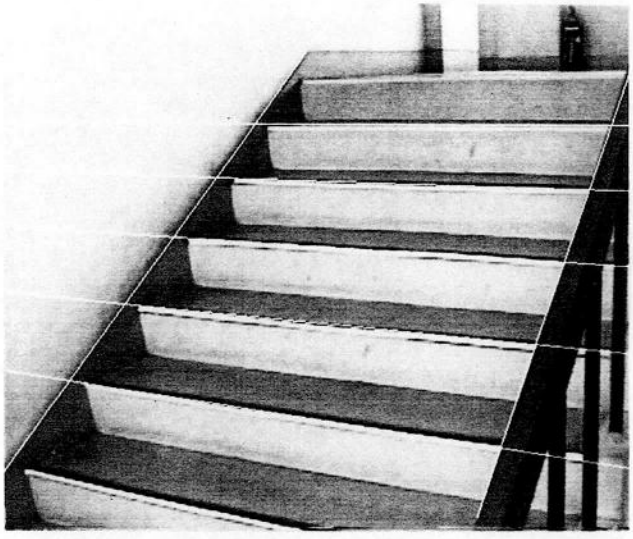


Photo.10.3 Testing environment

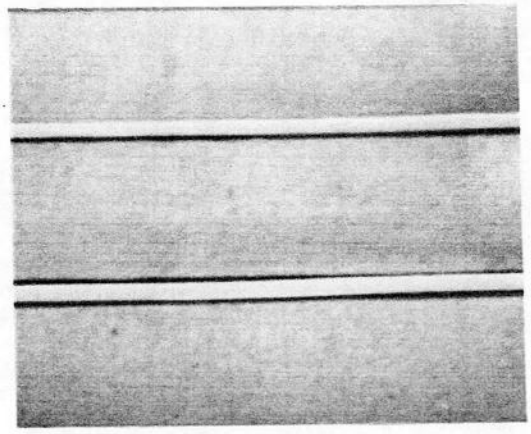


Photo.10.4　A pair of stairs

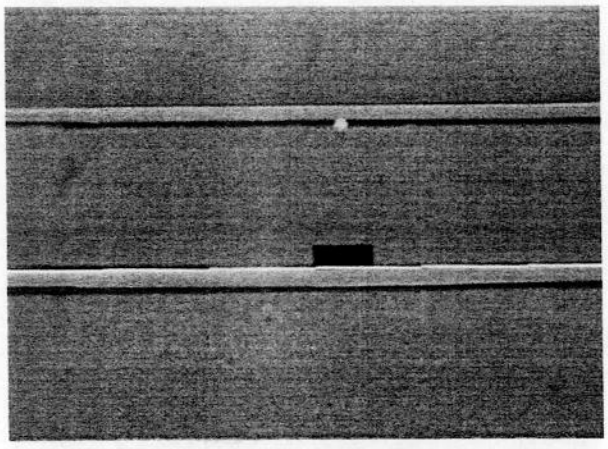


Photo.10.5 , 0[mm]

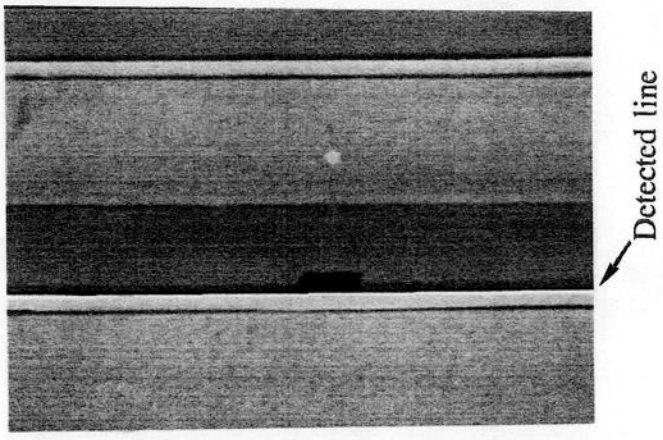


Photo.10.8 , 0[mm]

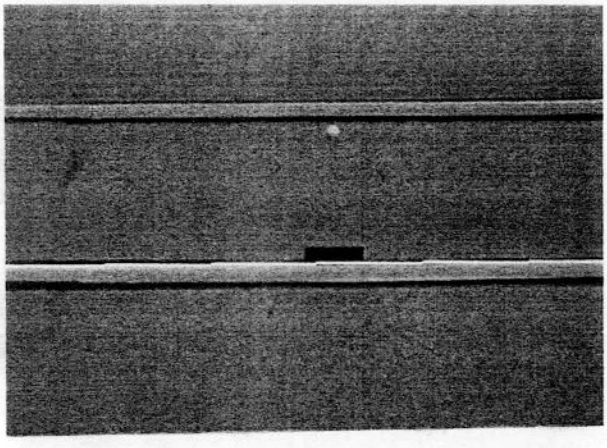


Photo.10.6 , 100[mm]

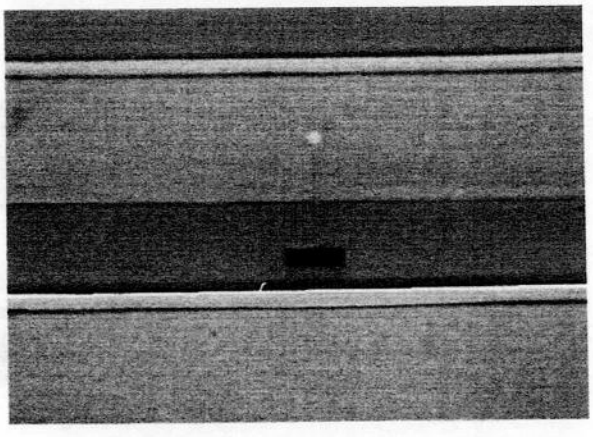


Photo.10.9 , 100[mm]

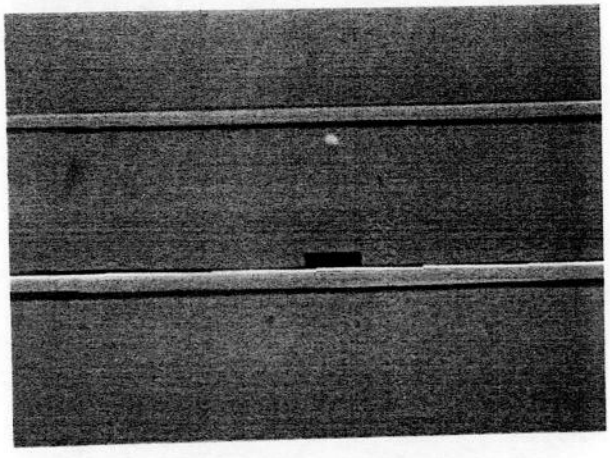


Photo.10.7 , 200[mm]

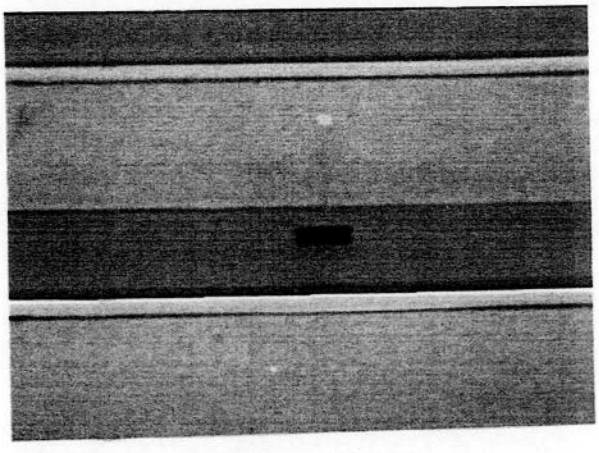


Photo.10.10 , 200[mm]

Table 10.1　Measured depth by image processing, actual depth and error

Position [mm]	P_1 [Pixel]	P_2 {pixel]	Depth h_r[mm]	True [mm]	Error [%]
0 [mm]	130	122	15	0	－－
100 [mm]	122	135	102.4	100	2.4
200 [mm]	119	157	209.0	200	4.5

奥行き100[mm],200[mm]のときは誤差は5[%]以内に収まっている．P_1,P_2の測定値が１画素でもずれるとh_rは約4[mm]の変化を生じ誤差感度が非常に高い．またカメラと被写体との距離が奥行きに対して十分大きくとれなかったことも誤差の原因になっている．これらのことを考慮すると表１０．１の結果は十分満足できるものと考える．

１０．４　道路上センターラインの検出

最近ＧＰＳを利用した自動車の自動ナビゲーションシステムが普及しつつある．これは地球の衛星軌道を回る２４個の人工衛星の内，最低３個以上の衛星からの電波を捉えて自車の位置を判定しディスプレイ上に表示するものでその位置分解能は数十メートル程度である．このようにして自車の位置や目的地までの距離，混雑状況を判断することが可能で運転者に多くの情報を与えてくれるのであるが最終的なハンドル操作は運転者に任されており無人での運転は不可能である．ところが最近になって，生産工場内での自動搬送車の考えを取り入れ，この無人運転の研究が開始されている．現在のところまだ初歩的な試験段階であるが，高速道路の車線中央に約２０[ｍ]間隔で強力な磁性体を埋め込み，これをマーカとして，車内に積んだ磁気センサがこれを感知し車体を車線の中央に位置するよう制御するものである．この実験は運輸省管轄下の研究機関で，高速道路の，完成はしているがまだ供用していない区間を使って行われている．

マーカとセンサの組み合わせは多くの種類があるが自動搬送車の場合の一つの問題点は上記のような磁気的センサの組み合わせの場合，それとは別に人間が目視確認できる進行方向の案内が必要になることである．つまり作業員と搬送車が混在する場合，搬送車がどのように進行するかを現場にいる人間が把握できなくなり，衝突等の事故が発生するのである．このため通路上には磁気マーカの配列に従って白線を描いている場合が多い．この点，通路上の白線，あるいはガイドラインを検出して，これをたどって運行する方法は単純で費用も安くて済む．但し表面の汚損の問題があり，現場が作業員が土足で歩くような場所には設置し難いという欠点がある．

さて本論文で検討を重ねているHough変換手法は，本来低品質の画像中から直線成分を検出することを目的に開発された．高速道路では一般にカーブを非常に緩くとってあり，

短い区間をとれば直線で近似できる．またほとんどの高速道路は片側二車線以上あり，車線の中央には分離ラインが描かれている．この分離ラインは断続的な直線で描かれており，通常の直線検出法では的確に捉えることが難しい．このような観点からHough変換をこの分離ライン，あるいは路側帯の分離ライン検出に利用できないかと考えた．これらが安定に検出可能であれば，両ラインの丁度中央を車が走行するように制御することが可能となる．なお以下の記述では分離ラインをセンターラインまたはサイドラインと呼ぶことにする．

このような装置を実際の走行システムに供用するには実時間でのHough変換が必須となる．本論文ではこの高速化の問題の検討に多くの頁数を費やしているが，まだハードウエア技術の進歩による更なる高速化がなされないと実用化は不可能である．ここでは実時間での直線検出実験ではなく，ビデオカメラで撮影してきた画像をもとにどの程度の正確さ，安定性をもって直線検出が可能かどうかを検討している．実験に使用したシステムは前節のものと同じである．

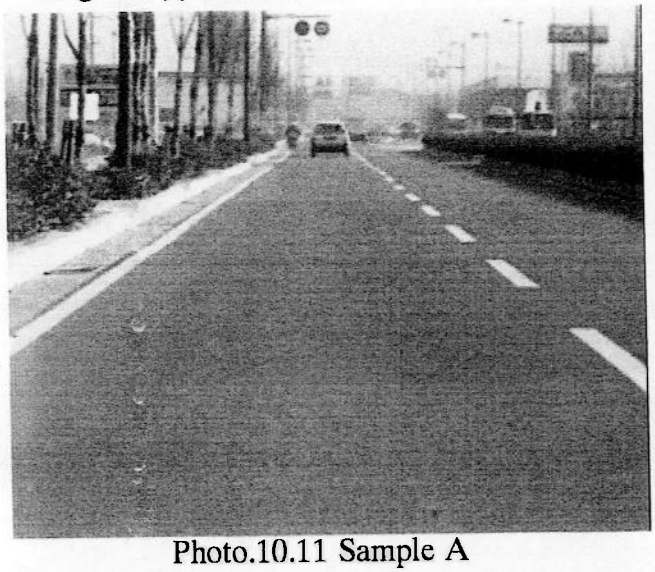


Photo.10.11 Sample A

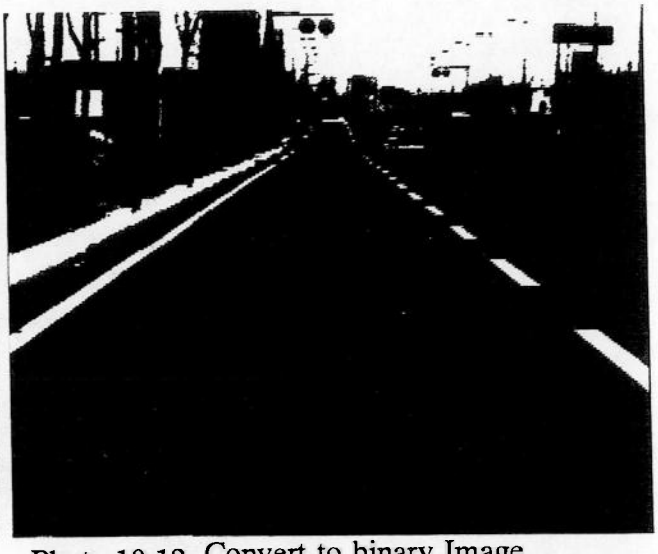


Photo.10.12 Convert to binary Image

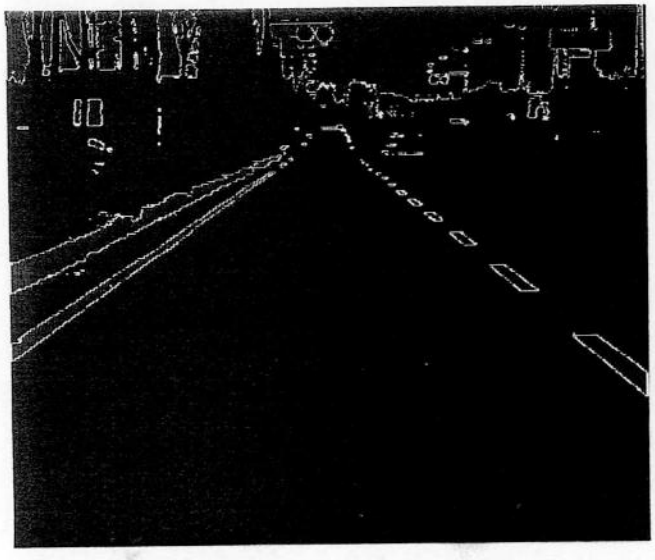


Photo.10.13 Edge enhanced

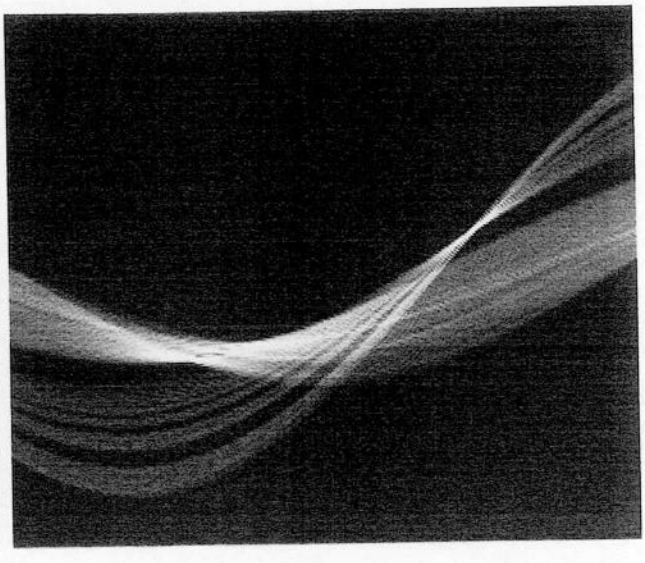


Photo.10.14 Hough plot

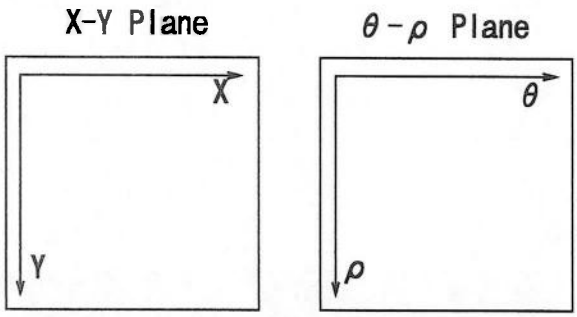


Fig.10.7 Direction of Y and ρ axis

Photo.10.15 Detected lines are Overlaied

１０．４．１　実験結果

供試する入力画像データは２つある．一つは，最近国道３号線の久留米市付近バイパスのため開通した高規格の見透視の良い直線道路である．これはセンターラインが破線でサイドラインが直線になっている．これをサンプルＡとする．今一つは久留米-柳川県道のカーブした付近で，追い越し禁止となっており，センターラインは黄色の連続カーブである．これをサンプルＢとする．ここは最近改修が行われセンターラインがはっきりしている．この２つのサンプルはハンディＶＴＲで撮影し，前記のシステムに入力した．

写真１０．１１はサンプルＡのほぼ直線の道路の原写真である．グレイスケールは０～２５５（０は黒，２５５は白）である．これを閾値１１４で２値化した．写真１０．１２にこれを示す．次に２次元微分を行いエッジを抽出した．写真１０．１３にこれを示す．これのエッジについてHough変換演算を行った．

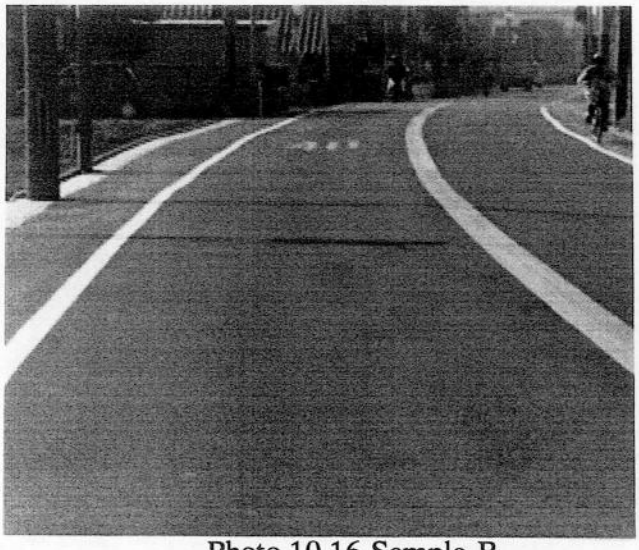


Photo.10.16 Sample B

photo.10.17 Convert to binary Image

その結果が写真１０．１４である．変換の対象となるエッジ上のサンプリング点は写真１０．１３の上１／４をカットした．サンプリング点の数が大きくなりすぎて時間がかかるのとθ-ρヒストグラムの度数値がオーバーフローするからである．サンプリング点の数は予想外に多く約3000点であった．Hough変換演算と度数累積に必要な時間は約２８[min]であった．最後に写真１０．１４について度数探索を行い，直線を検出して原画像上に重畳した．写真１０．１５にこれを示す．度数探索時間は探索範囲を絞った結果，３０[s]程であった．第１０．２節の記述に対して大幅な処理時間の増加となったがその理由は１０．４．２項の考察で示す．

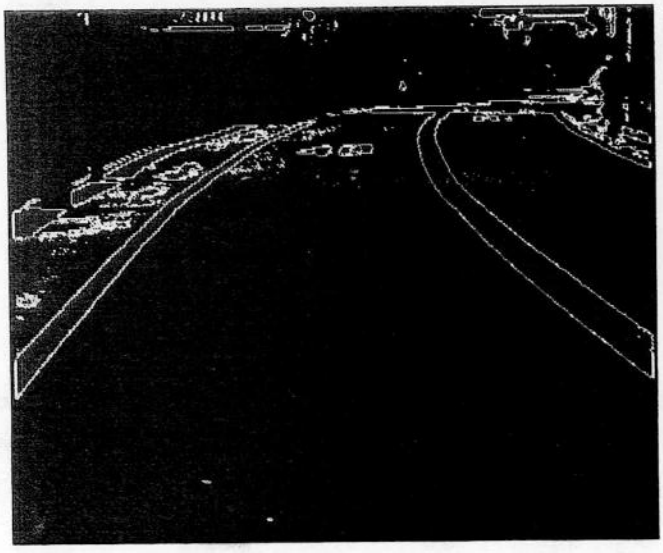


Photo.10.18 Edge enhanced

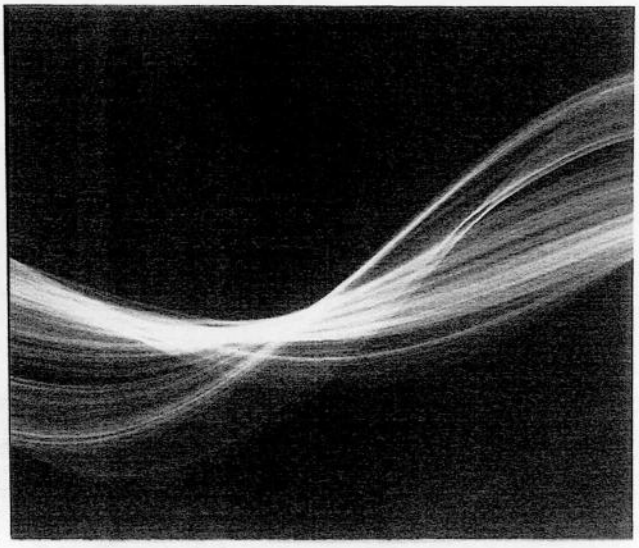


Photo.10.19 Hough plot

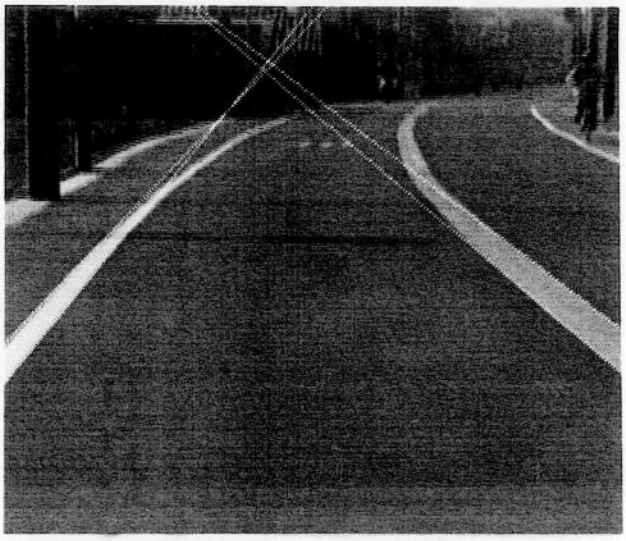


Photo.10.20 Detected lines are overlaied

Houghプロット及び度数探索は写真１０．１４のようなHough プロットの写真を撮るため本体のメインメモリ上ではなく画像 A/D 入力ボードのメモリ上で行った。

次にサンプルＢについて同様の実験を行った．写真を１０．１６～１０．２０に示すがその種類と順序はサンプルＡのときと全く同じである．２値化の閾値は140である．同じくサンプリング点の範囲を上部１／４程をカットした．その数は4500点程度になった．変換処理に要した時間は約４２[min]であった．度数探索の時間は同じく探索範囲を絞って４０[s]程度であった．上記いずれの場合にも度数探索は写真１０．１４，１０．１９から人間の目で判断し，その範囲を絞った．以上

挙げた写真１０．１１～１０．２０はY軸，ρ軸の方向が図１０．７のように上下反転している．これは使用したハードウエアとの関連でこうなった．従って写真１０．１４と写真１０．１９はこの関係を考慮して写真１０．１５，写真１０．２０中の直線と対応づけしなけらばならない．

１０．４．２　考察

（１）処理速度について

１０．２節で挙げた溶接部開先形状の検出の例と比較してHoughプロット、度数探索の時間は本節の例では極端に遅い．その原因は前項でも述べたが写真１０．１４，１０．１９を撮影する関係上別のシステムを使用したからである．まず前者がCPUにi80486 66[MHz]であるのに対し本節のシステムは i80286 12 [MHz]である。更に先にも述べたように度数探索範囲の目安をつけるためにHoughプロットと度数探索を画像入力ボードのビデオラム上で行ったからである．ビデオラム部は本体とバンク切り替え形式で接続されており，アクセスのためにアドレス切り替えのサブルーチンを介するので極端に速度が落ちてしまう．またサードパーティ製のボードでありメモリアクセスにクロックのウエイトが入っているようである．この種のパソコンはいわゆるDOSベースで処理を行う限り，画像処理用のメモリ空間が制限され既に限界状態になっている．作業用のワークアエリアとして１画面分で256[Kb]の領域が必須であり効率的な作業環境とは言えない．

以上の理由により処理速度を上げるため手作業の介入を数多く行った．まずHough変換サンプリング点の制限を行った．処理速度の向上のためが第一の理由であるが，今一つはθ-ρヒストグラム平面の１要素が１[byte]であるため累積可能な度数が255に制限されているのでサンプリング点を数多く与えることが困難だったことがその理由である．255をオーバーすると度数が０に戻ってしまう．度数探索の窓は写真１０．１４，１０．１９から目安を得た．

（２）処理結果についての考察

まずサンプルＡでは写真１０．１５を見るとセンターラインの破線が２本の直線として正確に捉えられている．これは写真１０．１４では右半分の高輝度集中点に対応している．この部分は写真では１点のように見えるがこの近傍を度数探索すると２個のピークがある．写真１０．１４の左側の高輝度集中点では３個のピークがある．この点は写真１０．１５のサイドラインの２本分に加えて歩道部との境界を示すコンクリートのブロックの並びの直線に対応している．これは目的外の直線の誤検出であり，好ましくないものである．

サンプルＢでは写真１０．１９の右半分の高輝度集中点が図１０．２０のセンターラインの２本の直線に対応している．直線部分が少ないので輝度も落ちている．それに対して写真１０．１８のサイドラインのエッジを見ると明らかに右側のエッジよりの輝度が高く見える．これは複数個のエッジ点がかたまっているからであり，写真１０．１９の中心から左にかけての高輝度部分に対応している．写真１０．１８ではサイドラインの左に，更に多くのサンプリング点があり，これが輝度を押し上げている理由ではないかと考えられる．

予想はしていたが曲線部分では撮像装置から最も近い位置からの線が直線として検出された．これは実用化の際には非常に有効である．車が渋滞すると自車から遠距離部分のラインを捉えることは難しくなるからである．

１０．５　結言

本章ではHough変換を用いた溶接開先形状の検出，３次元形状計測及び道路のセンターラインの検出の例を挙げた．第２番目の例はパターンマッチングによる自動計測の最も基礎となる事項である．パターンマッチングに関しては様々な手法が開発され実用に供されているが，この例でこの分野に深く立ち入ることは本論文の主題から離れることになるので行わなかった．そこでマッチングの容易な球状のターゲットとHough変換により検出される直線を基準点として利用し奥行きの検出を行った．測定結果の誤差は５［％］以内でありほぼ妥当な値である．

第１０．２及び１０．３節の問題がオフラインでの処理で十分であったのに対して１０．４節で取りあげた問題は将来的な応用としては実時間性を必要とするものである．１０．４．１項で述べたように特にサンプルＢのような場合にHoughプロットの様子がどうなるかという点を検討するため写真１０．１４，１０．１９を出力してみた．結果として逆に処理に多くの時間がかかってしまった点は問題であるが実用化の際は６章以降９章までの各手法を応用することにより，車載型の実時間Hough変換直線検出装置を開発することの目安が得られたと考える．

１１章　高速Hough変換と実時間形状計測を応用した矩形物体の検査

２次元図形の高速形状係数計測手法と高速化されたHough変換の手法を融合し，移動物体の形状パラメータを抽出して，形状が規定の寸法，形に適合しないものを摘出するシステムを開発した．特にベルトコンベアでの搬送中に，流れを中断することなく画像パラメータを計測するアルゴリズムを積極的に利用し，まず一般的な矩形形状物体からその形状パラメータを抽出する手法を述べる．次に実際の対象として養殖海苔の乾燥検査を採り上げ，実用システムとしての試作装置を制作した．この間，発生したいくつかの問題点とその解決法を明らかにする．最後に試作装置の評価を行い，今後の問題点を記す．

１１．１　緒言

筆者は第２，３，５章で述べたような，移動する物体を１次元のCCDイメージセンサで捉えた後２値化し，専用ハードウエアあるいは高速なソフトウエア処理により面積，重心，周囲長，形状係数を実時間で測定する新しい手法を開発してきた．また６章～１０章に示したようにHough変換に関して極座標型表参照方式，独立変数を分割して縮小した表を準備することによる分割縮小化表参照方式，連立漸化式による実時間処理に近い高速Hough変換の構成法を明らかにした．そこで、これらの手法を融合させ，直線で構成された図形の形状パラメータを高速に検出し，ベルトコンベア上を移動する物体の形状パラメータを製品規格と比較することにより，不良品を摘出することを試みた．

一次あるいは二次製品としての生産工程の途中で，製品は次段階の生産工程への受け渡しのため，ベルトコンベアで搬送されることが多い．コンベア上を流れる製品をラインイメージセンサで捉え，計算機の中に順次取り込めば製品のイメージがメモリ内に構成可能である．しかしこの間，計算機はアイドル状態になり，製品の通過直後から処理を開始することになる．このアイドル時間を効率よく活かすにはハードウエア的に，１，２走査線分のメモリ（ラインバッファ）を用意して，製品の通過中から演算処理を行う方が効率がよい．先に述べたいくつかの形状パラメータ検出手法は一方向からの走査で，その結果が得られるよう工夫されており，効果的に融合利用することにより，人手による手作業を省いた，効率の良い欠陥製品の選別が可能となる．本文ではまずこれまでの章で示した検査に応用する個別手法について説明する（１１．２節）．ついで一般的な矩形形状物体を対象として議論を進める（１１．３節）．次に実際的な応用例として養殖海苔の乾燥検査装置を試作し（１１．４節），その評価，問題点の抽出を行う（１１．５節）．なお本文で取り扱う画像データはすべてセンサからの出力をある閾値で切り分けた２値化データである．

１１．２　検査に応用する個別手法の説明

対象を矩形形状物体としているので，(a)矩形を構成する線分の長さ，その相対的な位置関係，(b)面積，重心，周囲長，形状係数などが形状パラメータとして挙げられる．(a)，(b)共に筆者が開発したアルゴリズムに基づいた専用のハードウエアを用意して実時間の計測を行う．これらは２章～１０章においてその詳細が示されている．ここではその概要

を説明し，関連する空間微分によるエッジ抽出などに触れる．

１１．２．１　実時間Hough変換について

Hough変換の手順についてはこれまでの章に何度か述べたがここで実際の応用例に関連させて簡単に振り返る．その手順は次の通りである．原画像X-Y平面内の画像エッジ上の一点(x, y)を次式で示されるHough変換式で$\theta-\rho$ヒストグラム平面に写像する．

$$\rho = x \cdot \cos\theta + y \cdot \sin\theta \qquad \text{---(11.1)}$$

そうすると(x, y)の１点に対して１本の正弦曲線が対応する．この正弦曲線の通過度数を$\theta-\rho$ヒストグラム平面の各要素に累積する．その結果としての累積度数のピーク点を探索しその点を(θ^*, ρ^*)とすれば原画像平面における直線の式はx, yを変数として

$$\rho^* = x \cdot \cos\theta^* + y \cdot \sin\theta^* \qquad \text{---(11.2)}$$

で表される．この関係を図１１．１に示す．この図は後で述べる応用例と関連させて対象である長方形を斜めに置いた状態で描いている．(a)は原画像空間に斜めに置かれた四角形のエッジである．(b)はこのエッジ上の各点についてHough変換式(11.1)により度数累積されたヒストグラムをドットの濃さで表現している．(a)図のA, B, C, D辺は同図(b)のA, B, C, D点と対応している．１１．４節に述べる応用例では図(a)の姿勢が非常に安定している．このため図(b)の度数ピーク点の位置及び度数の値も安定している．従って度数探索の窓を小さく絞ることが可能となった．これが本章での実時間Hough変換による直線検出が可能となった一つの理由である．

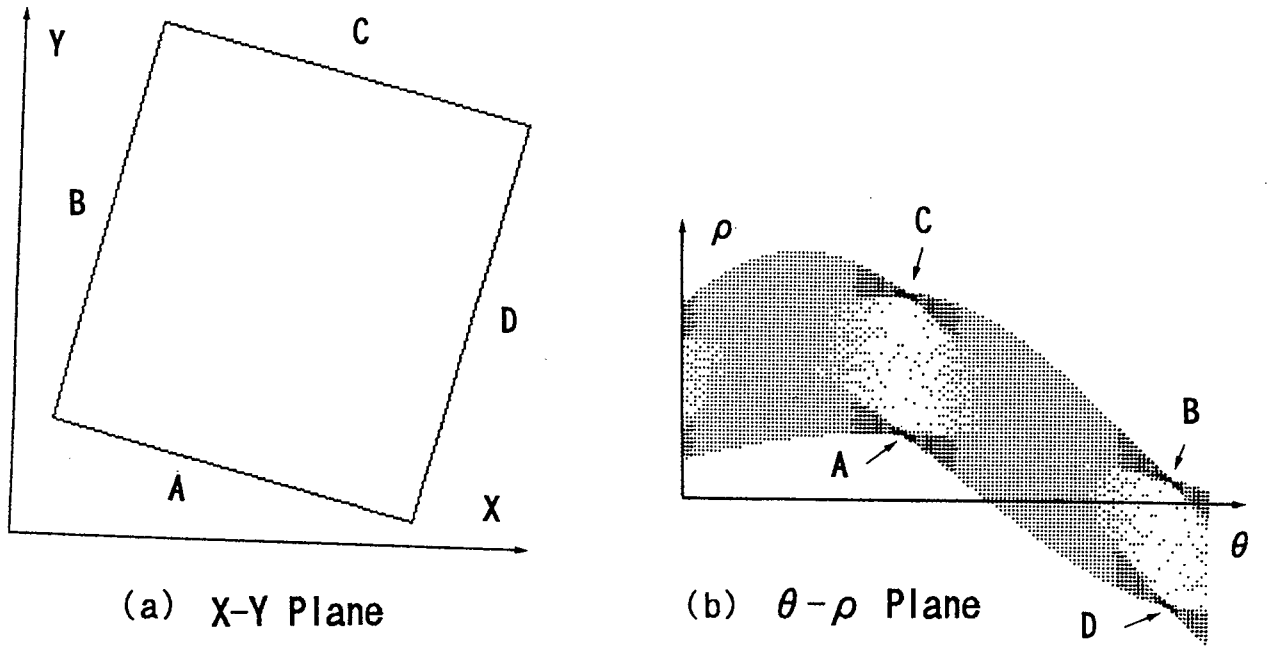


Fig.11.1　Relation between X-Y plane and $\theta - \rho$ plane

Hough変換演算を高速に実行する手法について７章～９章で，いくつかの方法を述べた．今回の応用例では筆者が独自に考案し，第７．３節で示した分割縮小化表参照方式を用いた[18]．この方式は比較的簡単なハードウエア構成で正弦曲線算出速度が$\theta-\rho$平面の１

点当たり50[ns]と高速だからである．

１１．２．２　実時間形状係数測定について

２章～５章で移動物体の周囲長、面積，形状計数，*X*方向重心，*Y*方向重心を１方向からの走査で測定する手順を述べた．通常は上記パラメータの前三者が規格との比較対象となるが、もし移動中の姿勢が一定しているならば，その重心も測定比較の対象となり得る．

ここで第３章にで示した手順を簡単に振り返る．図１１．２はイメージメモリの走査の様子を示している．常に２本の走査線を見ながら処理を進める．走査中，その走査線上の

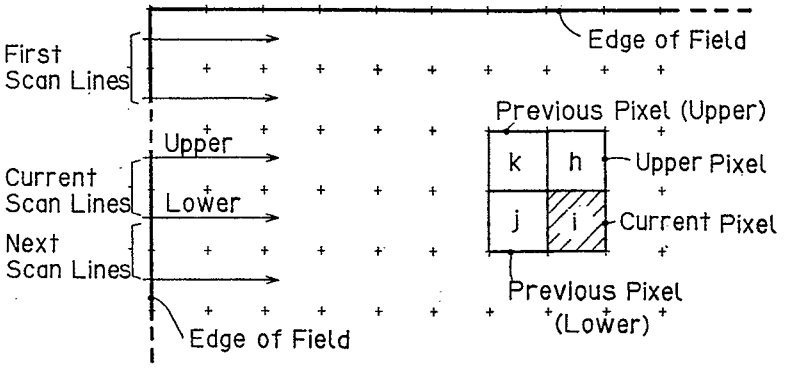


Fig.11.2 Scanning and pixels

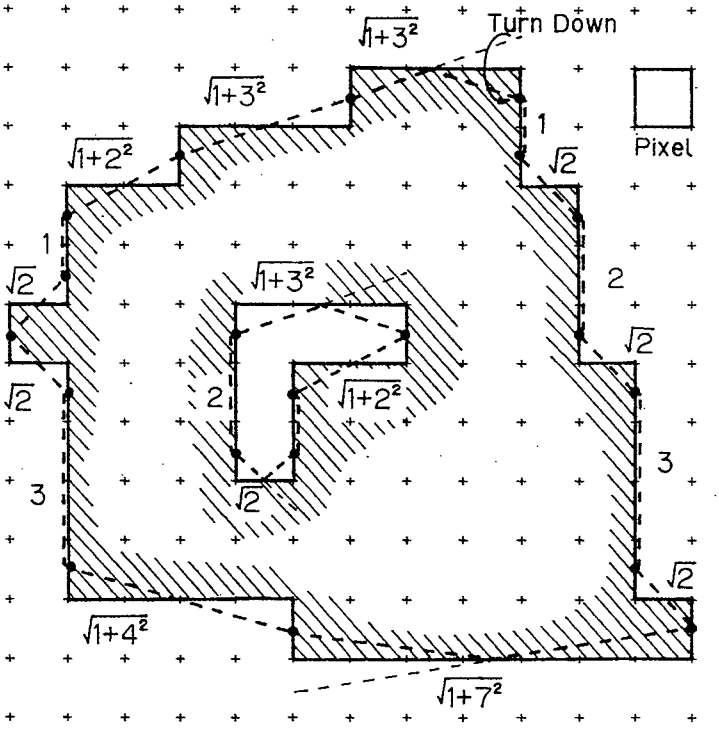


Fig.11.3 Example of perimeter calculation

４つの画素値が一定のパターンに一致したとき，輪郭長の計算が行われ累算部に加算される．この操作を画像平面全体にわたって行い最終的に累算部に総周囲長が求まる．この様子を図１１．３に示す．実線が原図形で、破線が求められる近似輪郭長である．この手順の特徴は縦長の長方形画素を利用する点にある．こうすることによって測定精度を大きく改善することすることが可能となる．この手順は図からも明らかなように走査線２本を監視しながら測定が可能なので，ラインイメージセンサと１走査線分のラインバッファを用意すれば処理が可能なことがその特長である．面積は単に画素値が"1"である画素個数の累算により簡単に求まる．重心については画素値が"1"である点のXまたはY座標を累算し，面積で除すれば求まる．形状係数Fは周囲長をL，面積をSとすると次式で与えられものとする．

$$F=L^2/(4\pi S)$$

１１．２．３　エッジ抽出のための空間微分について

画像輪郭を抽出するための手法は数多く発表されているが，ここではラインバッファの数ができるだけ少なくて済む方法として，次式で示される Robert のオペレータ[51]を採用した．

これは図１１．２のh, i, j, kを各位置の画素値とし輪郭値をmとすれば

$$m= |k-i|+|h-j| \qquad \text{---(11.3)}$$

これは式から判るように，注目する画素を中心に１つ前の画素，１走査線分先の画素とその一つ前の画素，計４個の画素を基に計算が行われる．必要なラインバッファの数は１である．ここで求まったサンプリング点座標は次の処理である Hough 変換のためにFIFOバッファに送り込まれる．

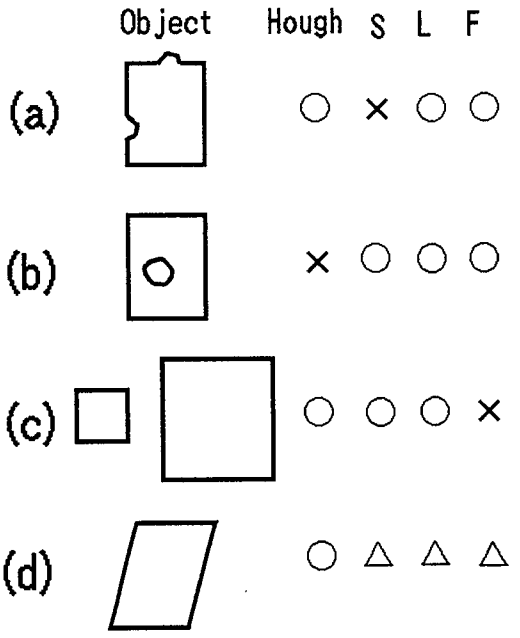


Fig.11.4 Defect detection by each method

１１．３ 一般的な図形を対象とした場合の検査について

１１．３．１ 対象の種類と検出手法の関係

本項では目標図形の欠陥の種類と，適用可能な図形パラメータとの関係を述べる．対象は次節との関係で、本来長方形であるべき図形をとりあげ，図１１．４を参照しながら考察する．

図では重心については省略した．図では"Hough"の欄がHough変換による直線検出でチェックできる項目，"S","L","F"はそれぞれ面積，周囲長，形状係数でチェックできる項目を示している．同図(a)では図形に凹部と凸部があり、面積は規格を満足するかもしれないのでチェックできず"×"である．その他は規格値から何らかのずれを生じ，判別可能である．これを○印で示している．(b)では図形中に「穴」があるが，輪郭は正常でHough変換では検出不可能である．（第２～４章で述べた周囲長測定の手順は「穴」の輪郭長も全輪郭長に加算される．）△は○×の中間を示している．(c)では面積の大小だけであり，形状係数は正常な場合と全く同じで，"F"での検出は不可能である．Hough変換の結果としては直線の位置，傾きは正常な値が得られるが，直線部の長さを示す度数値に変化を生じるのでチェックできる．但し，ピーク点位置θ^*，ρ^*の１点だけでは数値が不安定なので８近傍の度数値を加算して用いる．(d)の平行４辺形のケースは"S","L","F"にも変化が現れるが，実験的に直線位置の変化が最も顕著であることが判っているので"Hough"の項に○がついている．これらは全て，ゴー，ノーゴーチェックであり，実際の装置では図１１．４の○の項目についてチェックし，全てがゴー（良）のときのみ良品とする．これ以外は不良品として目視確認による検査を行う．

１１．３．２ 輪郭抽出とHough変換の並列処理

１１．２．３項に述べる方法で，移動中の図形から抽出された輪郭上の点の座標は，ファーストインファーストアウトバッファ(FIFO バッファ）に蓄え，（後掲，図１１．９ブロック図を参照）移動中もHough変換式による写像変換処理が並列に進行するようにした．この結果，図１１．１(b)に示すθ-ρヒストグラム形成の結果が製品の通過直後に得られることになる．

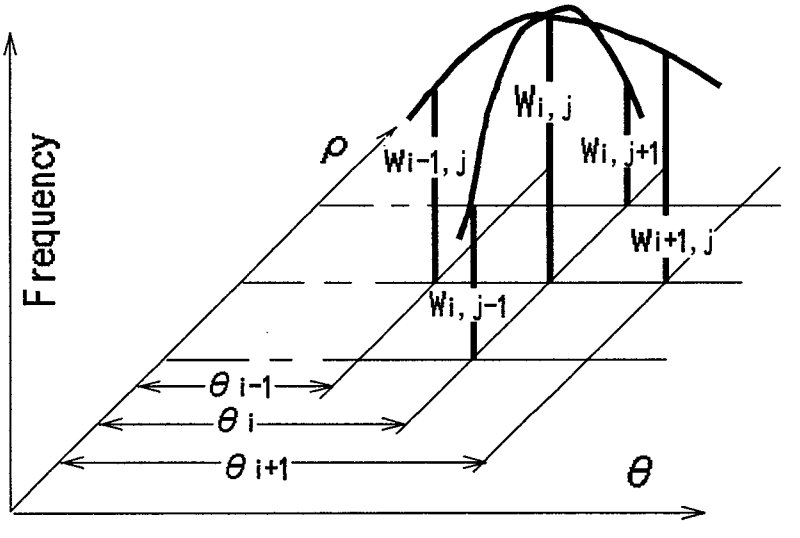


Fig.11.5 Interpolation of peek position

θ-ρヒストグラム平面はホストコンピュータのメインメモリとバンク切り替えで，直接接続されている．できあがったθ-ρヒストグラム平面の度数ピーク点探索がソフトウエア的に実行され，これより若干の時間遅れを持って度数ピークの探索が完了する．またその度数値は辺の長さに比例することから，原図形の大きさの規格値との比較選別が可能となる．

１１．３．３　ピーク点の位置決定と度数の計算

θ-ρ平面での度数探索の結果求まったピーク点位置を（θ_i，ρ_j）とし，その点の度数をW_iとする．ここで図１１．５に示すようにθ方向に次式で示される加重平均をとりθ_i^*とする．

$$\theta_i^* = (\theta_{i-1}W_{i-1} + \theta_i W_i + \theta_{i+1}W_{i+1}) \Big/ \sum_{k=i-1}^{i+1} W_k \qquad \text{---(11.4)}$$

同様にρ方向も処理する．

この処理はピーク探索に比し，必要な時間は僅かである．対象図形の単位時間当たりの処理個数を上げるためには走査速度（X方向）や図形の移動速度（Y方向）を上げねばならない．そうすると空間微分回路回路からFIFOバッファに送り込まれる単位時間当たりのサンプリング点座標の個数が多くなりバッファがオーバーフローすることがある．これを避けるためにはX方向分解能を落とさざるを得ない．そうするとピーク位置精度も低下するので前後の度数値との加重平均をとり精度の低下を避けている．

直線の長さに対応するθ-ρ平面ピークの度数はピーク点とその８近傍を加算した．

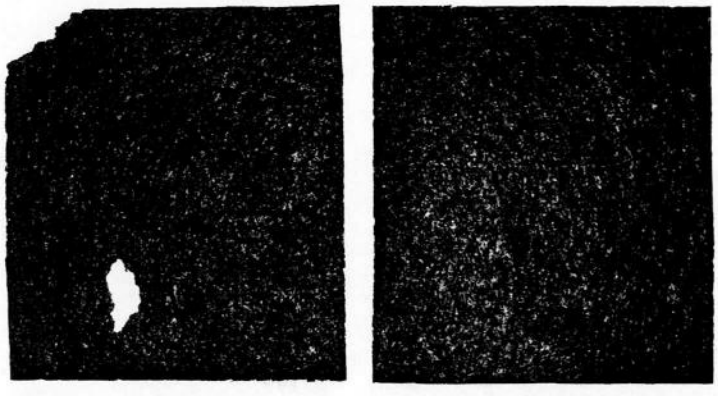


Photo. 11.1 Object Figure including defect and normal

１１．４　養殖海苔の製品検査への応用

完成した養殖海苔の製品はほぼ正方形で且つ薄い２次元平面とみなして良い．今までに述べた各手法を適用して，欠陥を持つ製品を摘出する装置を試作した．試作装置では１１．３節で述べた手法を取り入れた．

１１．４．１　対象の環境条件

商品として出荷される海苔は規格値で190 ×210[mm],色は艶があって黒いほどよく，適度の湿度を持つ必要がある．また端部にクラックがないこと，厚みが一様で，内部に空洞がないことが要求される．写真１１．１は欠陥部を持つ製品（左）と良品（右）を示している．左は一部が欠けている．また内部に空洞が存在する．海苔をすく環境は温度20±4[°]湿度60±15[%]に空調がなされている．ベルトコンベア上での移動速度は通常1300[mm/s]程度である．従ってラインイメージセンサで形状を捉えている時間は210/1300=0.16[s]=160[ms]]である．また単位時間当たりの枚数は２枚／秒である．

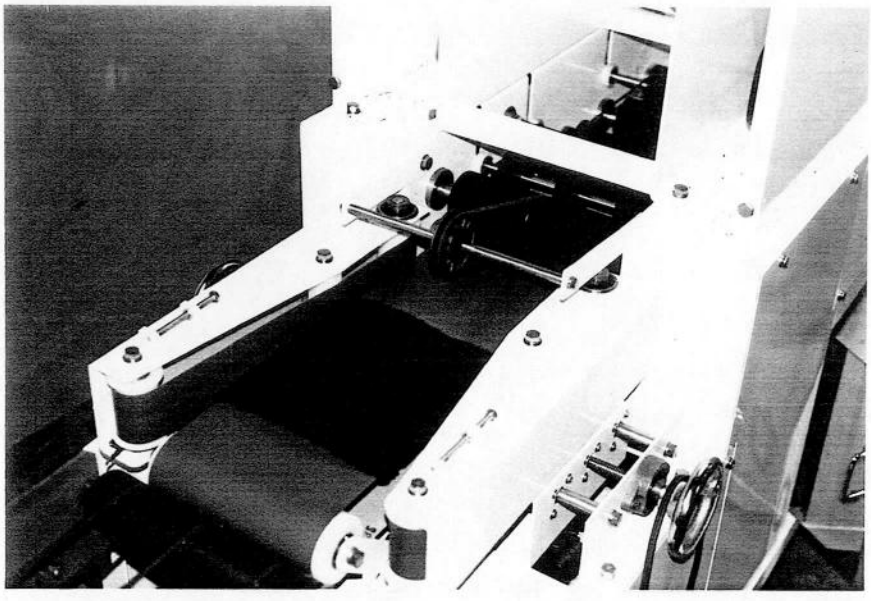


Photo. 11.2 Entrance of skew compensater

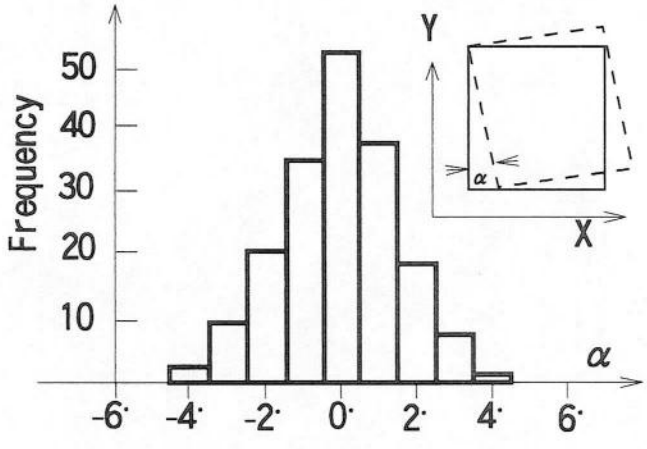


Fig.11.6 Histogram of error of position

海苔と次に来る海苔の間隔は距離にして1300/2-210=440[mm],時間にして338[ms]である．形状認識部における海苔の姿勢は、写真１１．２に示す姿勢補正部を通ることにより高い

精度で姿勢がそろう．このばらつきを図１１．６に示す．0[°]を中心に±4[°]に納まっている．合計185個のサンプルについて測定したが，標準偏差の３倍，3σは4.3[[°]であった．

１１．４．２　イメージセンサのセッティング

ラインメージセンサからの画像信号はＡ／Ｄ変換され，シェーディング補正回路を通った後，２値化される．ラインイメージセンサの線方向は図１１．７に示すように，対象の移動方向に対して斜め10°にセットされている．直角に配置すると図１１．８に示す如くθ-ρ平面の探索範囲が左右に分離し(B-B')，(D-D')，度数探索の手順が複雑になり、速度が落ちるからである．傾けて置いた場合の例である図１１．１(b)では度数ピークがはっきりと４個に分かれている．また傾斜させない場合，対象物体の前縁、後縁がセンサの下を通るときHough変換のサンプリング点が一度に多数発生しFIFOバッファがオーバーフローする．これを避けるためでもある．

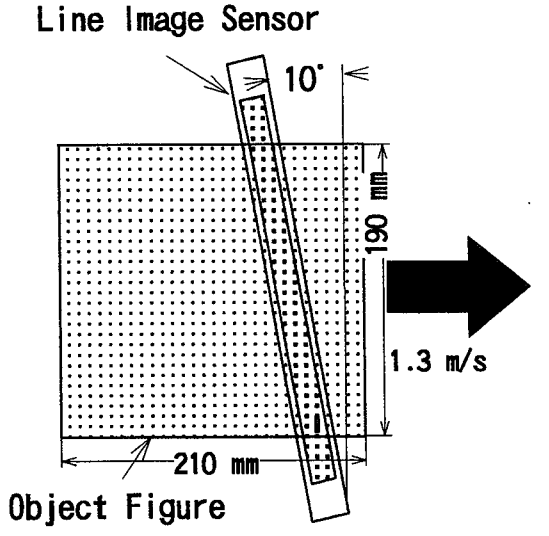


Fig.11.7 Setting of line image sensor to object figure

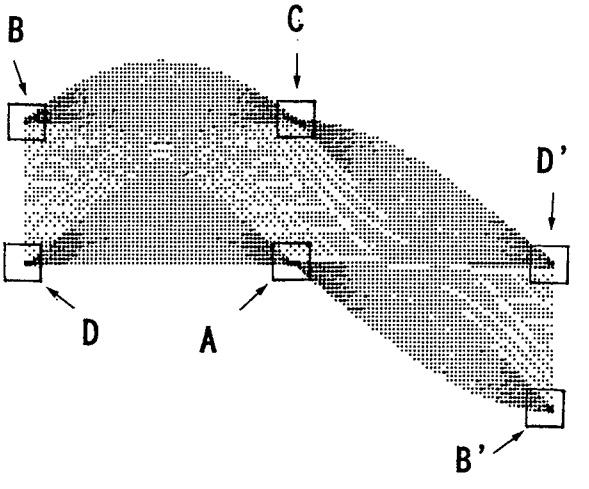


Fig.11.8 Histogram on $\theta - \rho$ plane

１１．４．３　専用ハードウエア

図１１．９に全体のブロック図を示す．図の左端にはＣＣＤラインイメージセンサと，対象物がセンサの下を通過する初めと終わりを検出するセンサが設けられている．　後者はX,Y方向のクロック同期に必要なものであり，それぞれの方向の座標の基準となる．その出力は重心測定のための累算器に接続される．その上には輝点の数を累算し面積を測定する累算器がある．これらの出力はバスを通してホストの計算機に接続されている．

イメージセンサの出力は一旦シェーディング補正回路を通り，２値化が行われる．その出力は１走査線分のラインバッファに入り１画素分の遅延回路を通して周囲長検出のための論理回路と，Hough変換のための空間微分回路に入る．

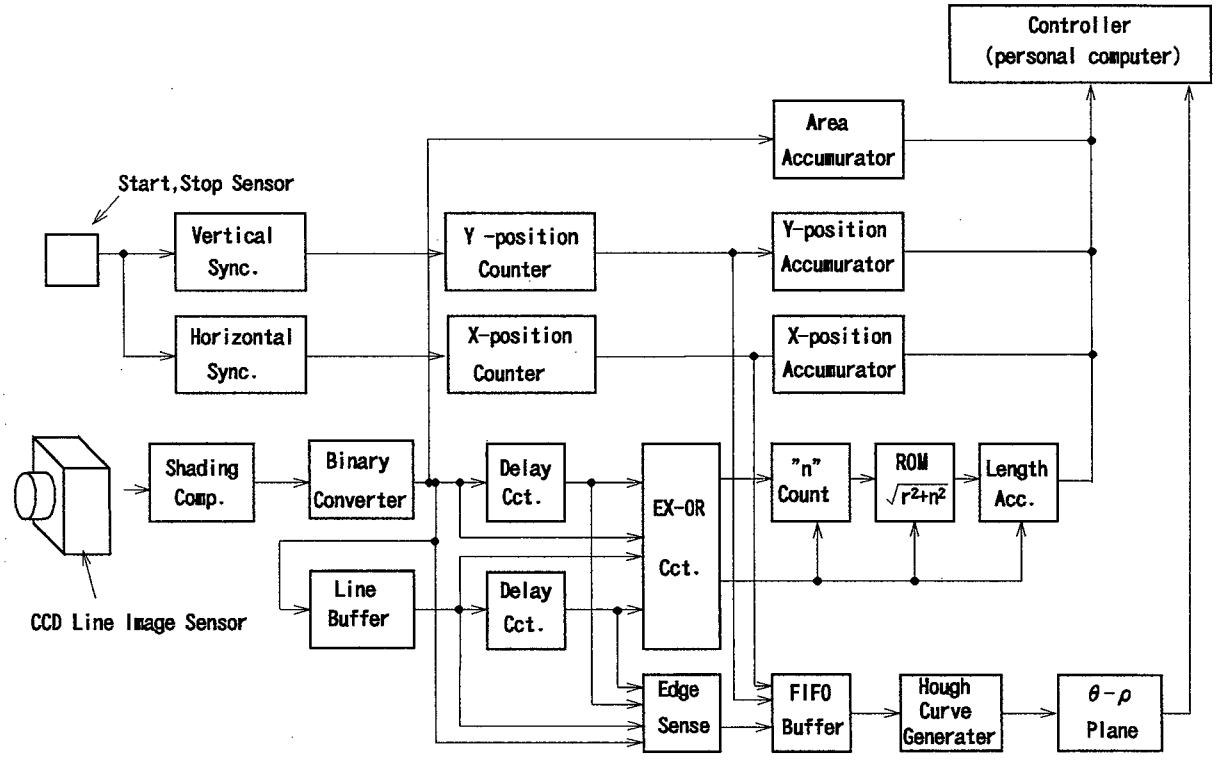


Fig.11.9 Block diagram of experimental hardware

排他的論理回路は輪郭の長さの計数と長さの計算部へ制御信号を発する．最下段の空間微分の回路の出力はFIFOバッファへ送り込まれ、Hough曲線発生部(ρ値計算部)へ接続される．その出力は本体のメインメモリとバンク切り替えで接続されていてθ-ρ平面での累積加算動作を行う．

１１．４．４　進行方向分解能とタイミング

イメージセンサは1024画素のものを1.33[MHz]で駆動し，実際には分解能を1/2に落として使用した．従って分解能512，クロックレート667[kHz]と考えて良い．

FIFOバッファは語長9[bit]×1024段のものをx,y座標用に２個用いた．

図１１．７の進行方向に直角な分解能（画素寸法）は走査速度の関係なしに一定である．対象図形がイメージセンサ全画素の75[%]を占めるとすると190[mm]/(512×0.75)≒0.5[mm],

従って３章に示したように画素の縦横比を最適値に近い2:1にとると2:1=1:0.5、すなわち製品が進行方向に1.0[mm]移動するとき１回の走査が完了すればよい．ここで走査時間をTとして、ベルトがa[mm/s]で進むなら1[mm]あたり$1/a$[s]かかるので$T=1/a$である．ここで１１．４．１項の速度の値を代入して$T=1/a=1/1300$[mm/s]$=0.77\times10^{-3}$[s]．従って１画素当たり0.77[ms]/512=1.5[μs]でこの逆数として先に挙げた走査クロックの値を定めている．

Hough変換はハードウエアで行われ$\theta-\rho$平面の一点あたり約50[ns]である(18)．画像輪郭点の数として400(≒512×0.75,進行と直角方向)×2(右辺と左辺)+200(進行方向)×2(上辺と下辺)=1200点を処理しなければならない．（上記200と400の数値は画素の縦横比が2:1になっているからである．実際の輪郭点の数は輪郭の斜め接続のため多少増える） Hough平面の$\theta-\rho$軸分解能をそれぞれ512とすると１図形当たりの変換時間は 50[ns]×1200×512=31[ms]となる．FIFOバッファを用いているので原画像の通過時間160[ms]に対して余裕があり，図形の通過直後に$\theta-\rho$平面の累積加算が終わる．この後$\theta-\rho$平面のピーク探索が始まる．図１．(b)のA～Dを探すのであるが，この部分はソフトウエア処理になるので速度向上のため探索窓を絞った．１１．４．１項で述べたように$3\sigma=4.3°$であるが余裕を持たせてθ方向には±6°にした．ρ方向はθ方向と同じ割合にとった．すなわち±6°/180°×512=±17[dot]となる．この結果、度数ピークの探索と１１．３．３項に述べた加重平均の計算に約40[ms]を要している．海苔と海苔の間隔は平均340[ms]あるので全く問題は無い．

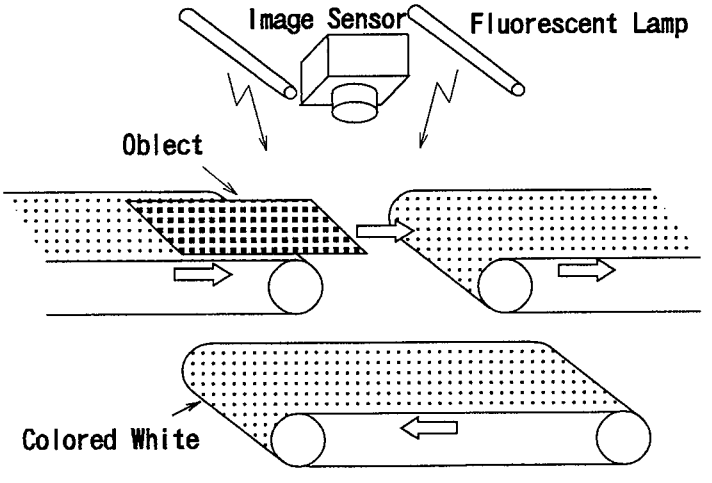


Fig.11.10 relational position of sensor, lightning and object

１１．５　試作機の構成と評価

１１．５．１　搬送検査部の配置

さまざまな試行錯誤の結果，コンベア，イメージセンサ，照明光源の配置，背景色用ベルトの配置は図１１．１０のように決めた．光源は蛍光灯である．普通のタングステンランプの発する波長帯の光では海苔を通過してしまうからである．またイメージセンサの両脇に配置したのは影をつくらないためである．下側に設けたベルトコンベアは白色に塗装

された背景色で，ゆっくり回転している．搬送中に欠けた海苔の砕片が落ち、イメージセンサが誤って感知しないようにするためである．回転しながらブラシで砕片をかき落している．この検査部は外光を遮断するため箱状のケースに入っており、その側面に電子回路部が納められている．写真１１．３はこの扉を開けた状態で撮影したものである．海苔の砕片が悪影響を及ぼさないよう、ホストとなる計算機は空調部屋とは別に設置されている．

１１．５．２　合否判定基準と評価

合格、不合格の閾値を定めるの必要があるのは検出された直線エッジの位置、傾き、θ－ρ平面のピーク位置近くでの度数の合計、面積、周囲長、形状係数である．いずれも規格値から５～２０％程の範囲でホストの計算機から指示できる．

最も発生頻度が高い，海苔の角部の欠け（写真１１．１左上隅）について定量的に調べてみる．正常な海苔は横寸法190[mm]、縦寸法210[mm]であるから図１１．４のＬは(190+210)×2=800[mm]と測定される．（画素の縦横比2:1とカメラセッティングの傾き10[°]についての補正を行った結果として800[mm]になる．以下同様）

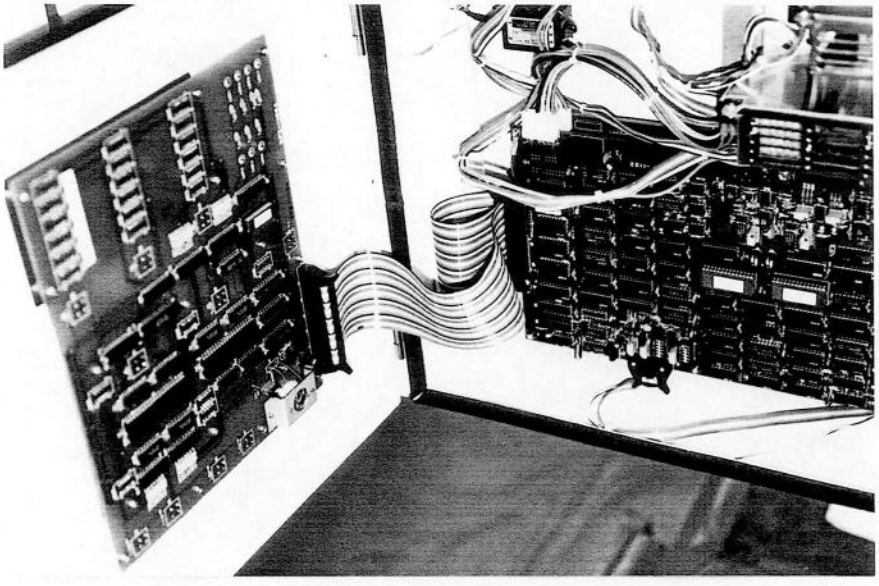


Photo. 11.3 Electronic circuit (lid opened)

同じくＳ，ＦはＳ=39900[mm²]、Ｆ=1.276である．Hough変換による累積度数（８近傍を加算した結果）は上辺，下辺，左辺，右辺の順に456[dot],456[dot],252[dot],252[dot]となった．画素の縦横比，対象図形の姿勢，微分オペレータの特性によりこの値が得られる．これを表１１．１の中欄に示す．測定のばらつきはいずれも３σ=３[%]前後である．これに対して写真１１．１の左のように左上部に20[mm]の欠けがあり上辺が170[mm],左辺が190[mm]の海苔に対しては表１１．１右欄の値が得られ（　）内に示す変化が現れる．最も顕著な変化があるのはHough変換での上辺，左辺の累積度数である．従ってこの種の欠陥に対してはHough変換の累積度数で不良品を摘出するのが最も効果的である．厳密にはこのような解析を欠陥の種類毎に行い、合否判定規格を定めるべきであるが実際にはそれぞれの値の最適規格値はトライアンドエラーで決めていった．規格を甘くすれば良品の中に不良品が混じる．厳しくすれば不良品と判定されたものの中に良品が入る．実際には

前者の割合を１００枚に１枚，後者の割合が４０枚に１枚程度になるよう設定している．後者の割合が多い理由は不良品としてはねられたものは，いずれにしろ，くず海苔としての需要に供するため，人手による再検査を必要とするから，そのときに取り出せばよいからである．またこの判定基準は製品市場での競り価格との問題で海苔製造業者の主観に左右されるので一概に決めがたい要素を含んでいる．このようにいくつかの問題点もあるが目視検査に必要な人手が１人分削減可能となり、実用面への応用の目的は十分達成されたと考える．

１１．５．３　その他の問題点

海苔くずの破片と水気，海苔が含む塩分の関係でホストの計算機を別室に置いたが，計算機のメインメモリの一部がバンク切り替えで接続され、検査部の電子回路部に組み込まれている．この間約2[m]程のケーブルで結ばれている．

Table 11.1 Comparison between shape parameter for normal laver and laver with defect

	Normal Laver	Laver with defect
L	800[mm]	760[mm](-5%)
S	39900[mm²]	39700[mm²](-5%)
F	1.28	1.16 (-9.3%)
Hough upper side	456[dot]	410[dot](-10%)
Hough lower side	456[dot]	456[dot] --
Hough left side	252[dot]	227[dot](-10%)
Hough right side	252[dot]	252[dot] --

このケーブルは厳重にシールドされているのであるがノイズの混入、波形のオーバーシュート、アンダーシュート，線間誘導などでの対策に多くの時間と労力が必要であった．実際の現象としてはプログラムの暴走となって現れた．このためホスト計算機のクロックレートを落としている．

１１．５．４　従来装置との比較検討

文献(22)ではコントローラとしてMC68000(モトローラ社製16bitCPU) を２個使用し*sin*

θ, $\cos\theta$の表を参照してHough変換のρ値を計算している．この文献では入力画像としてヒステリシス環線をとりあげているが，画像入力から直線検出までの処理に約1000[ms]を要している．著者等のシステムではこれに対応する数値は原画像の通過時間160[ms]とその後の度数探索、加重平均に必要な40[ms]をたした約200[ms]であり、著者等のシステムの方が速度が速い．軸分解能はほぼ同じである．さらに先の160[ms]は海苔の搬送速度から決まったもので測定系としてはまだ余裕がある．

文献(65)ではHough変換を利用して無人走行ロボットの進行方向検出を行っている。このシステムではコントローラとしてi80286を用い、画像入力からHough変換を経てハンドル量の制御までの時間を約100[ms]で行っている．このシステムではX-Y平面の軸分割数を64×64としており、我々のシステムの512×512に比較すると精度は非常に低い。

汎用の画像処理装置としてIX-C100（サンクス（株）製）がある．この装置はHough変換を用いず，移動中の物体がラインイメージセンサ下を通過するときの映像信号そのものを利用して図形の直径、間隔、通過度数等のパラメータを計測できる．処理速度は測定対象により異なるが測定対象の通過後からの演算処理時間が著者等のシステムの40[ms]に対し1～2.5[ms]程度でこの方が速い．しかし直線成分の検出などの能力は持っていない．

１１．６　結言

２章～５章で述べた高速な周囲長，形状係数の測定法と６～１０章で述べた実時間Hough変換の研究成果を融合し，一般的な矩形輪郭を持つ図形の形状パラメータ検出の方法を述べた．またこれを海苔の乾燥検査という実際的な例に応用して好結果を得た．この装置は一部の改良を経て平成7年10月頃から製品の出荷が始まっている．

本文で述べた方法は養殖海苔の乾燥検査だけではなく矩形形状をもつそのほかの製品、例えば瓦（第５章に応用例を挙げた），タイルなどの検査にも適用可能でその応用範囲は大きい．

第１２章　総括

本研究は２値化された図形データからいくつかの形状パラメータを高速に検出することを目的としたもので、これまで計算機上でソフトウエア処理されていた各種図形パラメータの検出手法を外部ハードウエアを用いて高速に実行できる様アルゴリズムの改善、新しい提案，対ソフトウエアインターフェースの最適化などを行い処理速度，精度，ハードウエア構成の容易さなどの観点から検討を行ったものである．

本研究で得られた成果は各章毎に詳述したがここではその内容を総括して述べることにする．

第１章ではこの分野に関連する従来の研究成果を概観するとともに本研究の必要性と研究の位置付けを行った．

第２章では第３．４章で述べる縦長画素採用の効果について概説し，画像入力と並行して処理を行うことによる高速化について触れた．また２値化画像処理の基本となる対象図形の２値化パターンの生成について詳しく検討を行った．対象図形の輪郭を画素レベルで微視的に見るとほぼ直線で近似できる．しかし対象図形が画素サイズに対して相対的に小さい場合直線では不満足で円弧で近似する必要性が生じる場合がある．そこでこの章では画素を点ではなく空間的な広がりを持った正方形として捉え，この正方形画素の上を対象図形がどのように覆うかを基準として画素値を決定する方法を採った．

図形輪郭が画素寸法に対して相対的に長い直線であるとき位置的にサンプル２値化されてできる２値化デジタル画像の輪郭は階段状の図形となる．ここではある直線の傾きが与えられたときに生じる階段状のステップは$1:n$と$1:n+1$または$n:1$と$n+1:1$の２種類であることを示し，それぞれの個数を直線の傾きをもとに算出可能なことを示した．また図形輪郭が曲線であってこれが画素寸法に対し直線と見なせないような場合について円弧が画素を横断する状況を分類しそれぞれの場合について当該画素の画素値を決定する手順を導いた．

コンピュータ画像処理最初の入力となる撮像装置からの信号には必ず電気，光学的なノイズが重畳している．このようなノイズが形状パラメータ抽出に与える影響について検討した．

第３章では図形の周囲長，形状係数を高速に計測する方法について考察した．従来この分野では微小図形から十分大きな図形について高精度に周囲長，形状係数を測定する”３画素ベクトル法”と呼称される手法があったがアルゴリズムが複雑で外付けのハードウエア上で実行するには不向きであった．そこでラスタスキャニング形式の画像が入力される時点で１走査線分のバッファを用意し、隣り合う２本の走査線間の４個の画素値をもとに周囲長を連続的に累算測定する新しいアルゴリズムを開発した．このアルゴリズムは高速で且つハードウエア化も容易である．しかし垂直方向の直線エッジに対して測定誤差が多かった．この問題を画素の形状を縦長の長方形にする事によって解決した．画素の形状を縦長にするには水平方向分解能を上げるか垂直方向分解能を落とさなければならない．ソフトウエアシミュレーションによれば後者の方法を採った場合にも面分解能が低下しているにもかかわらず周囲長測定精度は向上するという結果を得た．しかし無制限に垂直分解能を落とすとその方向の細かい凹凸を見落とす可能性があり最適値が存在する．同じくソ

フトウエアシミュレーションの結果として画素の縦横比は垂直方向対水平方向の長さ比が約2:1のときに最も安定的に精度が高いことが判った．また円弧を含む様々な図形に対してこの手法を適用して実験を行った結果、幾何学的な図形に対するよりも自然界に存在する一般的な自由曲線に対して誤差が少ないという結果を得た．この結果に基づき本アルゴリズムをハードウエア上にインプリメントした．撮像装置は市販の計測用モノクロカメラを使用しインターレースモードの１フィールドスキャンの時間で周囲長を含む面積、重心、形状計数を求めることが可能である．画素を縦長に採るには１～２走査線分のデータを飛び越すことにより実現した．結果的にソフトウエアシミュレーションの結果とほぼ一致するデータを得ることができ，アルゴリズムの正当性が実地に確認できた．図形の回転移動、直線移動に対する微小図形の形状係数測定についても３画素ベクトル法に匹敵する性能を得た．

第４章では外付けハードウエアを用いずに計算機内部で前記アルゴリズムを実行することを検討した．この場合正方形画素を前提としているので縦横比を連続的に変化させることは不可能である．しかし前述のように縦横比は2:1の近くに最適比が存在することが確認されているので水平方向の画素並びを１段分スキップする事により長方形画素を実現した．スキップは比較のために０～３の範囲で行ってデータを採取した．ノイズの影響のない理想的な実験環境を作るため入力画像は計算機のメモリイメージとして作成した．画素寸法に比して十分長い直線エッジについて，スキップを行って周囲長を累算していく際には複雑な階段状のステップを伴った部分輪郭長を考慮しなけらばならない．本論文ではこれらについても生成されるステップの種類とその個数を数学的に厳密に導いた．画素寸法に対して相対的に微小な正方形については一辺が画素寸法の20倍程度まで、円弧については半径が画素寸法の10倍近くのものまで周囲長測定精度を１％以下に押さえることが可能なことを示した．

第５章では第３．４章で述べた手順の応用実施例としてコロニアル瓦を対象としてディジタル測長の実験を行った．結果として測長誤差１[%]以下の値を得た．面積の測定誤差は0.3[%]前後であった．

第６章ではHough変換による直線検出の過程をいくつかのステップに分けて考察し，それぞれのステップにおける問題点とその解決法について述べた．また第７，８章で提案する新しいアルゴリズムについて概説した．

第７章、８章ではHough変換のρ値計算の高速化についての研究を行った．第７章では最も構成が単純で処理速度も速い表参照方式を取り上げた．この方法では本文でも述べたがソフトウエア上で実行する場合、メモリリソースを大量につぎ込んで多次元（３次元）の表を用意してもあまり速度の向上がみられないと報告されていたものである．この原因を調査してみると３次元の表の目的要素を特定するためにコンパイラ上では現れない固定小数点の乗算、加算が含まれていることが主な原因であることが判った．その点ハードウエア上で表索引を行えばこのような問題は皆無である．但し３次元の表（約128[Mbyte]）を用意することはＩＣメモリ一個あたりの容量が年を追って倍加しつつある現在でもあまりにも大きな負担である．そこで２次元の範囲内で表を用意して容量を低減し且つ速度が速い２種類の表参照方式，極座標型表参照方式及び分割縮小化表参照方式を示した．

前者はHough原式に含まれる正弦と余弦の項を振幅と初期位相の項を持つ正弦関数一個

にまとめ、表の初期位相に対応した位置からインクリメンタルに表を参照する方式である．この方式は表のためのＲＯＭとしては市販の安価なＩＣ一個でよい．但し現在市場に出回っているＲＯＭはアクセス時間がスタティックＲＡＭに比し遅い．試作装置はＲＯＭで構成したがこの部分をＳＲＡＭで構成すれば速度は倍以上に向上可能である．但し表内容のプリロードが必要である．ＲＯＭのアクセス単位は8ビット／語であるが符号ビットをうまく活用することによって９ビットに相当するρ軸分解能を得ている．

後者はθ-ρ平面の一本の曲線に対応する原図形上の縁辺点座標(x,y)を２進数で表示したときの変数ビットをいくつかに分割してそれぞれ表を用意し、参照して取り出したデータを加算する方式である．これは前述の方法よりも更にＲＯＭ容量の低減が可能で処理速度も速い．しかしHough変換による図形パラメータ検出手順全体としてみるとこの後の処理として、θ-ρヒストグラム形成処理があり、この回路の速度がＳＲＡＭを用いても一定以上の速度が得られない．ρ値の算出と度数累積加算はパイプライン的に実行可能であるからいずれか一方の処理時間が全体の処理速度を決定する．この方法ではθ-ρ平面の一点あたり50[ns]を得ている．これはこの後処理であるθ-ρヒストグラム探索の速度にもよるが実時間性の高い速度である．

第８章では連立漸化式による高速Hough変換について考察を行った．本論文ではこの連立漸化式の基本となる行列を正弦関数の加法定理から導き、それが時計回りの座標回転行列であることを示した．この基本式を近似式で置き換えることによって高速Hough変換に用いられるいくつかの漸化式が誘導されることを述べた．これらの漸化式は基本的に乗算をシフト演算に置き換えることによってに演算の高速化を計っている点が共通している．またシフト演算形式にはできないが座標回転行列そのものを表現する漸化式も示した．これらはそれぞれ特徴があって演算速度と精度とは互いにトレードオフの関係にある．従って実際面へのアプリケーションの要求する速度あるいは精度に応じてそれぞれ使い分けをしなければならない．

これらの近似漸化式はそれぞれ異なる一般解を持っている．これらの一般解を行列の理論によりそれぞれ導出することができた．この一般解の形から定性的に誤差と処理速度の推定を行った結果とソフトウエアシミュレーションの結果は一致した．これらの漸化式を用いたρ値計算はソフトウエア上で実行する場合も十分高速性を備えているが実時間処理への応用を考えハードウエア化の検討を行った．ハードウエアで取り扱う数値の形式は小数部を有する符号付き固定小数点データとした．浮動小数点形式にした場合はＤＳＰなどの浮動小数点演算機構を持ったデバイスにたよらざるを得ず回路が複雑になると同時に高価になるからである．

以上に述べた漸化式は時計回りの座標回転行列を使うと言う意味で共通している．これに対して時計回りの行列とこれからπ／2ずれた位置から反時計方向回りの座標回転行列を併用することによって４重並列にρ値を発生することができる４重並列漸化式を考案した．そして４重並列化の副次的効果として精度が向上することが確認できた．またハフプロットした際のピーク点の分離という現象もなくなることが判った．この４重並列化漸化式については速度、精度ともに最も有望な手法と考え，ハードウエア化について詳細な検討を行った．現時点ではこのハードウエアはまだ完成していないがＰＧＡ(Programmable Gate Array)などを用いれば数個のＩＣで構成できる可能性がある．むしろこのρ値発生

回路の後段に接続される度数累積加算回路の構成の方が複雑になるであろう．その理由はこの回路はかなりの大きさのθ-ρ平面用のメモリを含んでおりＰＧＡで構成するには難があるからである．

このようにHough変換のρ値発生度数累算部が高速化されると相対的に問題となるのがθ-ρ平面内の度数ピーク点の検出処理である．これについては度数累算ハードウエアにおいて度数加算のために当該番地の値を読み出したとき、一定値以上の度数をもつ要素のアドレスをハードウエア上のバッファに蓄積して置き、計算機内でのピーク探索の目安にするという方法を提案している．

第９章では同じくHough変換の高速化の問題を取り扱っているがこれまでの方式と性質を異にしている．ここで提案している方法はρ値が計算された後の度数累算の高速化に関するものである．ρ値の発生方法はこれまでに挙げたいずれの方法でもかまわない．この発生されたρ値情報をD/A変換してアナログ信号とする．そしてこれをブラウン管の垂直軸に印加する．水平軸にはθ方向のアドレスを同じくD/A変換したものを与える．そうすればブラウン管面上にHough曲線が描かれる．これを光学系を通してＣＣＤイメージセンサの感光面に投影する．ＣＣＤイメージセンサの単位素子はフォトダイオードと積分コンデンサにより構成されておりこのコンデンサにHough曲線の通過度数が電荷としてアナログ的に蓄積される．このように光学系を用いて空間的にρ-θ平面のアドレッシングを行おうとするものである．そしてθ-ρ平面の蓄積電荷量をＡ／Ｄ変換してホスト計算機メモリに転送する際、累積度数が大きい部分のアドレスをバッファに蓄積しておきホストコンピュータ上での度数ピーク探索の指針とする点については８章で述べた考え方と同じである．ただ最終的な結果としての度数をチェックしている点が異なる．

実験に使用したオシロスコープとＣＣＤカメラを組み合わせた方式ではオシロスコープ各軸の非直線性、光学系の歪みなどの補正に多くの労力を費やした．もし実際にこの原理を実用に供しようと考えるならば同章の末尾に述べたＬＣＤ等を利用したＩＣ化回路が現実的である．この考え方は将来実現するであろう光立体電子回路の応用実施例として非常に有望である．

第１０章ではHough変換の応用実施例を挙げた．一つはレーザービーム光切断法による溶接部位の形状検出である．今一つはHough変換を利用して立体視の基準線を検出し、対象物体の奥行きを計算する例である．更に道路上センターラインの検出の例を挙げた．

第１１章では高速な周囲長、面積、形状係数測定ハードウエアと、同じく高速なHough変換回路を利用し養殖海苔の良、不良検査に応用した例をとりあげた．Hough変換はアカデミックな学界で話題が大きくとりあげられたほどには実際の応用例が少ない．これはそれらの内の数少ない実施例の一つである．ここでは撮像装置として１次元のＣＣＤイメージセンサを使用している．ベルトコンベア上を移動する対象物体を画像取り込みを行いながらパラメータ算出の演算を平行して実行していく．Hough変換のサンプリング点は時間的にランダムに発生するから、この部分にはFIFO (First In First Out)バッファを組み込んでいる．またθ-ρ平面の度数探索に関しても対象物体の姿勢安定性が高いことを利用し探索窓を絞っている．このような様々な試行錯誤の結果、装置の実用化が可能となった．

第１２章では本研究の成果を総括し結論とした．

謝辞

本研究は，大阪大学接合科学研究所　井上勝敬教授の懇切な御指導と，暖かい御鞭撻を頂いて遂行し得たものである．久留米工業高等専門学校　電気工学科主任　大淵豊教授には公私にわたる数々の有益な御教示並びに御討論を頂いた．また大阪大学工学部　生産加工システム講座　荒井栄司教授，並びに同大学　生産加工プロセス・機器工学講座　仲田周次教授には本論文の査読の労をとって頂いた．ここに厚く御礼申し上げます．

なお本研究は大阪大学接合科学研究所（旧溶接工学研究所）の共同研究員制度にもとずく研究の成果であることを記します．お世話になった同研究所事務部の皆様に感謝の意を表します

付録１．画素の隠蔽面積を基準として２値化図形を得る手順

（１．１）直線エッジの２値化

２．３．１項　補題(1)の証明

図ａ．１のように傾き 1:n に２値化されたとすると
画素Ｅは"0"であるから直線はＥの中心より下を通る。
画素Ｆは"1"であるから直線はＦの中心か,それより上を通る。
画素Ｇは"0"であるから直線はＧの中心より下を通る。
画素Ｈは"1"であるから直線はＨの中心か,それより上を通る。
以上の直線の存在範囲$\Delta\theta$を図ではドットのハッチングで示している．したがって傾き最大の直線①を考えると　　b/a ＜ $1/(n-1)$
傾き最小の直線②を考えると　　$1/(n+1)$ ＜ b/a
となりそれぞれの逆数をとって式(2.2)が得られる．

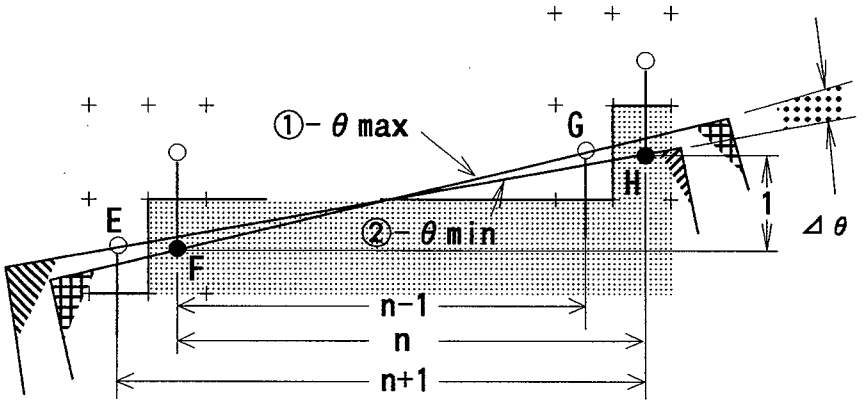


Fig.a.1　Binary image and original straight lines　(1:n)

２．３．１項　補題(2)の証明

傾き45°をこえる直線については原点を通る45°の直線で折り返して考えればよい．図ａ．２のように傾き n :1に２値化されたとすると
画素Ｆは"0"であるから直線はＦの中心より下を通る。
画素Ｅは"1"であるから直線はＥの中心か,それより上を通る。
画素Ｈは"0"であるから直線はＨの中心より下を通る。
画素Ｇは"1"であるから直線はＧの中心か,それより上を通る。
以上の直線の存在範囲$\Delta\theta$を図ではドットハッチで示している．
したがって傾き最大の直線①を考えると　　b/a ＜ (n+1)
傾き最小の直線②を考えると　　(n-1) ＜ b/a
となり式(2.3)が得られる．

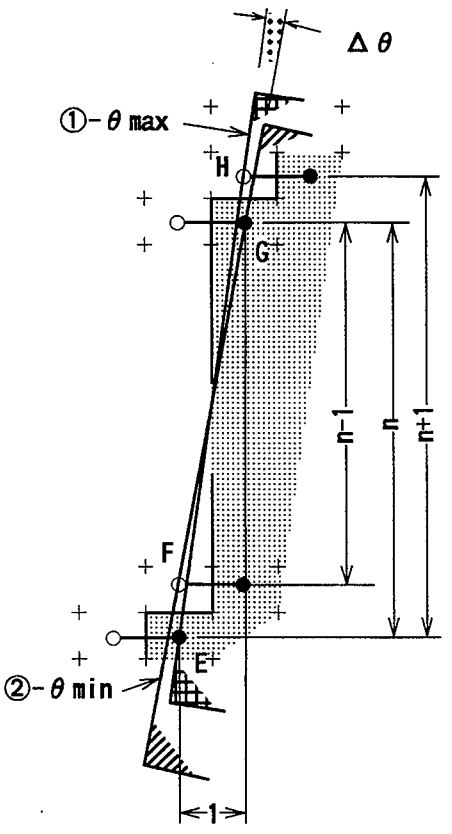


Fig.a.2 Binary image and original straight lines (n:1)

２．３．１項　定理(1)の証明

n が与えられたとき 1:n のステップを生じる，傾きの逆数 a/b（ここでは除算の結果で考える．このことを明示するため：ではなく/を用いる．その値は実数である．）の集合 P_nは式(2.2)によって数直線上で 図ａ．３ ② の如く示される．両端は空である．同様にして 1:n+2，1:n+1，1:n-1に対応する $P_{n+2}, P_{n+1}, P_{n-1}$ は④，③，①で示される．P_i は一般項である．図より数直線上の特定の一点は P_i の内の１つに属するか，または隣接する P_i、P_{i+1}に属することがわかる．したがって傾き b:a が与えられたときに生じるステップは　”1:iだけ”　か、　”1:i と 1:i+1”　のいずれかである．図中破線で示した a/b は P_nのみに属し前者に対応する．また a'/b' はP_nとP_{n+1}に属し後者に対応する．いずれの場合も数直線上の位置関係から n は a/bまたは a'/b' の整数部である．

次に X方向への移動量 a と、Y方向への移動量 b を考えると

$$a=np+(n+1)q$$

$$b=p+q$$

これより p, q を求めると式(2.4)、(2.5)が得られる．q=0 のときは”1:iだけ” に対応し 1:n のステップのみを生じる．

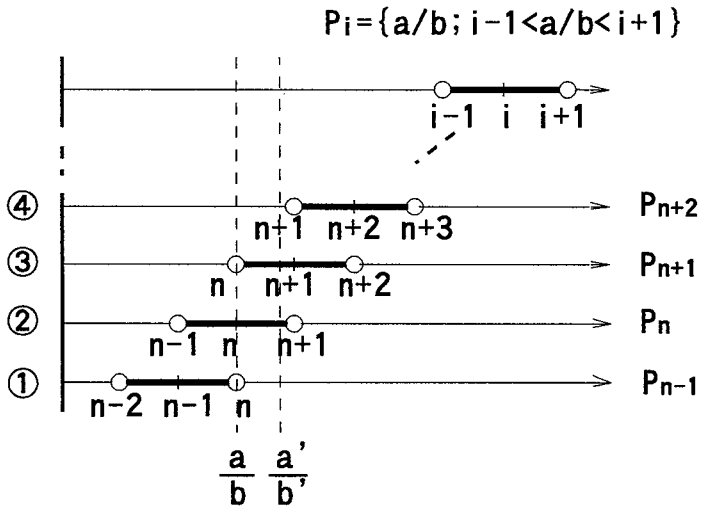


Fig.a.3 A set of reciprocal of a/b on numerical line

（１．２）円弧エッジの２値化

２．３．１項　定理(2)の証明

次のように補題(2)を用いて証明できる．n が与えられたとき n:1 のステップを生じる，傾き b/a（ここでは除算の結果で考える．このことを明示するため：ではなく/を用いる．その値は実数である．）の集合 P_nは 式(2.3)によって数直線上で 図ａ．４ ② の如く示される．両端は空である．同様にして n+2:1，n+1:1，n-1:1に対応する $P_{n+2}, P_{n+1}, P_{n-1}$ は④，③，①で示される． P_i は一般項である．

図より数直線上の特定の一点は P_i の内の１つに属するか，または隣接する P_i、P_{i+1} に属することがわかる．したがって傾き b:a が与えられたときに生じるステップは ”i:1だけ” か、 ”i:1 と i+1:1” のいずれかである．図中破線で示した b/a は P_nのみに属し前者に対応する．また b'/a' はP_nとP_{n+1}に属し後者に対応する．いずれの場合も数直線上の位置関係から n は b/a または b'/a' の整数部である．

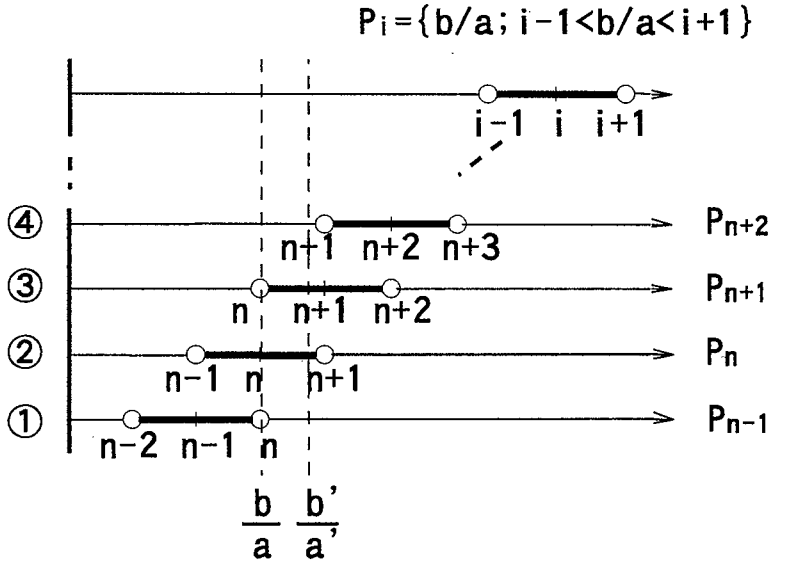


Fig.a.4 A set of reciprocal of b/a on numerical line

次に X方向への移動量 a と、Y方向への移動量 b を考えると

$$a=p+q$$

$$b=np+(n+1)q$$

これより p,q を求めると式(2.6)、(2.7)が得られる．$q=0$ のときは"i:1だけ" に対応し n:1 のステップのみを生じる．

２．３．１項の積分を求める手順の詳細

S_3について

$$S_3=\int_{X_1}^{X_2}\int_{y=y_1}^{y=\sqrt{r^2-(X-X_c)^2}+y_c} dydx = \int_{X_1}^{X_2}(\sqrt{r^2-(X-X_c)^2}+y_c-y_1)dX$$

ここで積分公式

$$\int\sqrt{r^2-(X-X_c)^2}\,dX=\frac{1}{2}(X-X_c)\sqrt{r^2-(X-X_c)^2}+r^2\sin^{-1}\frac{X-X_c}{r}\Big\}+C \qquad ---(a.1)$$

を用いて

$$S_3=\Big[\frac{1}{2}\Big\{(X-X_c)\sqrt{r^2-(X-X_c)^2}+r^2\sin^{-1}\frac{X-X_c}{r}\Big\}+(y_c-y_1)X\Big]_{X_1}^{X_2}$$

$$=\frac{1}{2}r^2\Big\{\sin^{-1}\frac{X_2-X_c}{r}-\sin^{-1}\frac{X_1-X_c}{r}\Big\}$$

$$+\frac{1}{2}\Big\{(X_2-X_c)\sqrt{r^2-(X_2-X_c)^2}-(X_1-X_c)\sqrt{r^2-(X_1-X_c)^2}\Big\}+(y_c-y_1)(X_2-X_1)$$

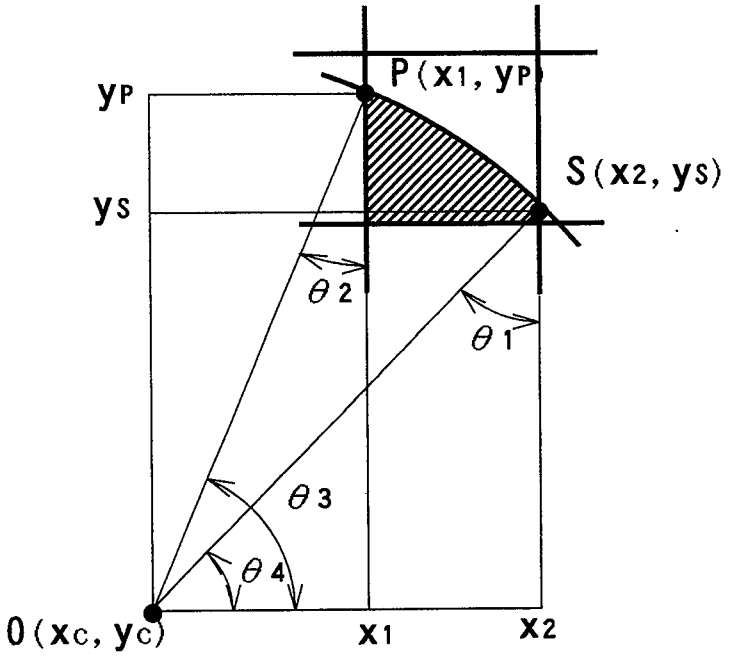


Fig.a.5 Intersection of arc and pixel (case 3)

図ａ．５より $\sqrt{r^2-(X_2-X_c)^2}=y_s-y_c$ ， $\sqrt{r^2-(X_1-X_c)^2}=y_p-y_c$

$\theta_1=sin^{-1}\dfrac{X_2-X_c}{r}$ ， $\theta_2=sin^{-1}\dfrac{X_1-X_c}{r}$ ， $\theta_3=cos^{-1}\dfrac{X_1-X_c}{r}$ ， $\theta_4=cos^{-1}\dfrac{X_2-X_c}{r}$

また $\theta_1-\theta_2=\theta_3-\theta_4$ であるから

$$S_3=\frac{1}{2}r^2\{cos^{-1}\frac{X_1-X_c}{r}-cos^{-1}\frac{X_2-X_c}{r}\}+\frac{1}{2}\{(X_2-X_c)(y_s-y_c)-(X_1-X_c)(y_p-y_c)\}-(y_1-y_c)(X_2-X_1) \quad \text{---(a.2)}$$

S_4について

$$S_4=\int_{X_1}^{X_R}\int_{y=y_1}^{y=\sqrt{r^2-(X-X_c)^2}+y_c}dydX=\int_{X_1}^{X_R}(\sqrt{r^2-(X-X_c)^2}+y_c-y_1)dX$$

ここで積分公式(a.1)を用いて

$$S_4=\left[\frac{1}{2}\{(X-X_c)\sqrt{r^2-(X-X_c)^2}+r^2sin^{-1}\frac{X-X_c}{r}\}+(y_c-y_1)X\right]_{X_1}^{X_R}$$

$$=\frac{1}{2}r^2\{sin^{-1}\frac{X_R-X_c}{r}-sin^{-1}\frac{X_1-X_c}{r}\}+\frac{1}{2}\{(X_R-X_c)\sqrt{r^2-(X_R-X_c)^2}-(X_1-X_c)\sqrt{r^2-(X_1-X_c)^2}\}+(y_c-y_1)(X_R-X_1)$$

図ａ．６より $\sqrt{r^2-(X_R-X_c)^2}=y_1-y_c$ ， $\sqrt{r^2-(X_1-X_c)^2}=y_p-y_c$

$\theta_1=sin^{-1}\dfrac{X_R-X_c}{r}$ ， $\theta_2=sin^{-1}\dfrac{X_1-X_c}{r}$ ， $\theta_3=cos^{-1}\dfrac{X_1-X_c}{r}$ ， $\theta_4=sin^{-1}\dfrac{y_1-y_c}{r}$

また $\theta_1-\theta_2=\theta_3-\theta_4$ であるから

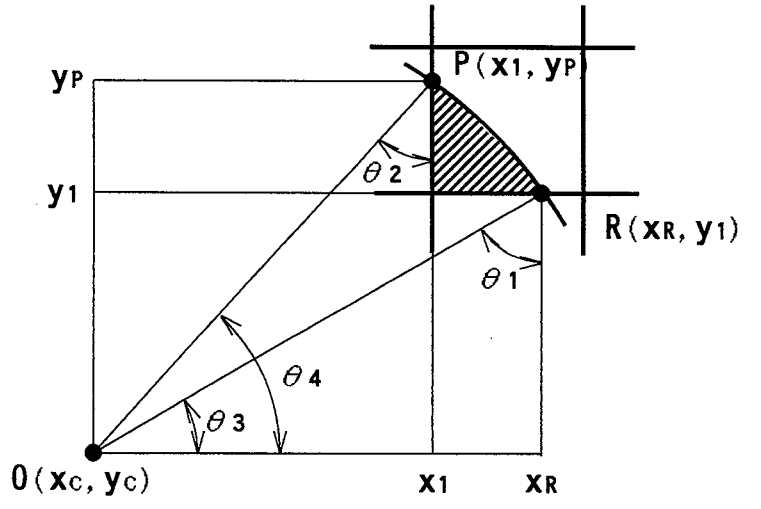


Fig.a.6 Intersection of arc and pixel (case 4)

$$S_4=\frac{1}{2}r^2\{cos^{-1}\frac{X_1-X_c}{r}-sin^{-1}\frac{y_1-y_c}{r}\}$$

$$+\frac{1}{2}(X_R y_1 - X_R y_c - X_c y_1 + X_c y_c - X_1 y_p + X_1 y_c + X_c y_p - X_c y_c + 2y_c X_R - 2X_1 y_c - 2y_1 X_R + 2y_1 X_1)$$

$$= \frac{1}{2}r^2\{\cos^{-1}\frac{X_1 - X_c}{r} - \sin^{-1}\frac{y_1 - y_c}{r}\}$$

$$-\frac{1}{2}\{(X_1 - X_c)(y_p - y_1) + (X_R - X_1)(y_1 - y_c)\} \qquad \text{---(a.3)}$$

S_8について

$$S_8 = \int_{y_1}^{y_2}\int_{X=X_1}^{X=\sqrt{r^2-(y-y_c)^2}+X_c} dxdy = \int_{y_1}^{y_2}(\sqrt{r^2-(y-y_c)^2} + y_c - y_1)dx$$

ここで積分公式(a.1)を用いて

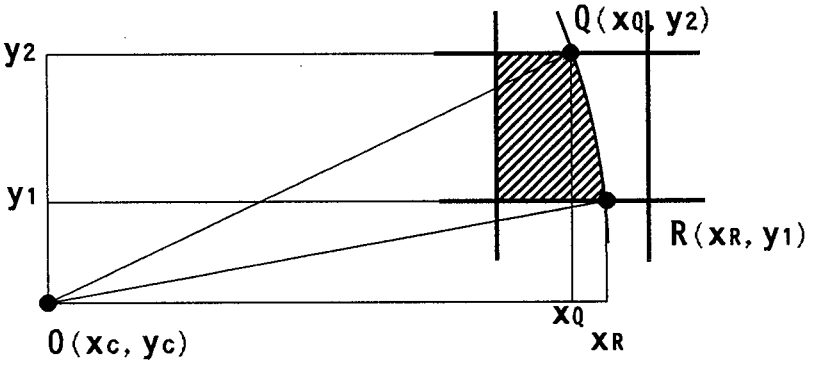


Fig.a.7 Intersection of arc and pixel (case 8)

$$S_8 = \left[\frac{1}{2}\{(y-y_c)\sqrt{r^2-(y-y_c)^2} + r^2\sin^{-1}\frac{y-y_c}{r} + (X_c - X_1)y\right]_{y_1}^{y_2}$$

$$= \frac{1}{2}r^2\{\sin^{-1}\frac{y_2 - y_c}{r} - \sin^{-1}\frac{y_1 - y_c}{r}\}$$

$$+\frac{1}{2}\{(y_2 - y_c)\sqrt{r^2-(y_2-y_c)^2} - (y_1 - y_c)\sqrt{r^2-(y_1-y_c)^2}\} + (X_c - X_1)(y_2 - y_1)$$

図a．7より $\sqrt{r^2-(y_2-y_c)^2} = X_Q - X_c$ ， $\sqrt{r^2-(y_1-y_c)^2} = X_R - X_c$

$$S_8 = \frac{1}{2}r^2\{\sin^{-1}\frac{y_2 - y_c}{r} - \sin^{-1}\frac{y_1 - y_c}{r}\}$$

$$+\frac{1}{2}\{(y_2 - y_c)(X_Q - X_c) - (y_1 - y_c)(X_R - X_c)\} + (X_c - X_1)(y_2 - y_1) \qquad \text{---(a.4)}$$

となる．

付録２．行列の理論による漸化式の一般解の導出

（２．１）座標回転行列について

８．２節で連立漸化式を用いてρ値を逐次計算する方法を述べ，式(8.10)を導いた．これを下に再掲する．

$$\boldsymbol{\rho}_n=\begin{Bmatrix}\cos\Delta & \sin\Delta\\ -\sin\Delta & \cos\Delta\end{Bmatrix}\cdot\boldsymbol{\rho}_{n-1}=\mathbf{A}\cdot\boldsymbol{\rho}_{n-1}$$

この式右辺の係数となっている行列

$$\mathbf{A}=\begin{pmatrix}\cos\Delta & \sin\Delta\\ -\sin\Delta & \cos\Delta\end{pmatrix}$$

はベクトル$\boldsymbol{\rho}_{n-1}$の座標$(\rho_{1,n-1},\ \rho_{2,n-1})^T$を時計方向回りに$\Delta$だけ回転した値を$\boldsymbol{\rho}_n$に与える座標回転行列である．この種の行列の性質として

$$\mathbf{A}^n=\begin{pmatrix}\cos\Delta & \sin\Delta\\ -\sin\Delta & \cos\Delta\end{pmatrix}^n=\begin{pmatrix}\cos(n\cdot\Delta) & \sin(n\cdot\Delta)\\ -\sin(n\cdot\Delta) & \cos(n\cdot\Delta)\end{pmatrix}\qquad\text{---(a.5)}$$

である．

（２．２）並列型漸化式

与えられた漸化式は

$$\boldsymbol{\rho}_n=\begin{bmatrix}1 & \Delta\\ -\Delta & 1\end{bmatrix}\cdot\boldsymbol{\rho}_{n-1}=\mathbf{A}\cdot\boldsymbol{\rho}_{n-1}\quad\text{但し}\mathbf{A}=\begin{bmatrix}1 & \Delta\\ -\Delta & 1\end{bmatrix}\equiv\mathbf{A}(\Delta)$$

ここで新しい変数δ_1を用意し図a.8の如くΔとの関係を定めれば

$$\cos\delta_1=\frac{1}{\sqrt{1+\Delta^2}}\qquad\sin\delta_1=\frac{\Delta}{\sqrt{1+\Delta^2}}\qquad\delta_1=\tan^{-1}\Delta$$

従って$\mathbf{A}(\Delta)$は

$$\mathbf{A}(\Delta)=\sqrt{1+\Delta^2}\cdot\begin{bmatrix}\cos\delta_1 & \sin\delta_1\\ -\sin\delta_1 & \cos\delta_1\end{bmatrix}=\sqrt{1+\Delta^2}\cdot\mathbf{A}(\delta_1)$$

但し

$$\mathbf{A}(\delta_1)=\begin{bmatrix}\cos\delta_1 & \sin\delta_1\\ -\sin\delta_1 & \cos\delta_1\end{bmatrix}$$

ここで式(a.5)を用いて

$$\boldsymbol{\rho}_n=\mathbf{A}(\Delta)\cdot\boldsymbol{\rho}_{n-1}=\mathbf{A}(\Delta)^n\cdot\boldsymbol{\rho}_0=\{\sqrt{1+\Delta^2}\cdot\mathbf{A}(\delta_1)\}^n\cdot\boldsymbol{\rho}_0$$

$$=(\sqrt{1+\Delta^2})^n\cdot\begin{bmatrix}\cos\delta_1 & \sin\delta_1\\ -\sin\delta_1 & \cos\delta_1\end{bmatrix}^n\cdot\boldsymbol{\rho}_0$$

$$=(\sqrt{1+\Delta^2})^n\cdot\begin{bmatrix}\cos(n\cdot\delta_1) & \sin(n\cdot\delta_1)\\ -\sin(n\cdot\delta_1) & \cos(n\cdot\delta_1)\end{bmatrix}\cdot\boldsymbol{\rho}_0$$

これを展開して初期値$(\rho_{1,0},\rho_{2,0})^T=(x,y)^T$を与えれば

$$\rho_{1,n}=(\sqrt{1+\Delta^2})^n\cdot\{x\cdot\cos(n\cdot\delta_1)+y\cdot\sin(n\cdot\delta_1)\}$$

$$\rho_{2,n}=(\sqrt{1+\Delta^2})^n\cdot\{-x\cdot\sin(n\cdot\delta_1)+y\cdot\cos(n\cdot\delta_1)\}$$

となり式(8.29),(8.30)が得られた．ここで　$\delta_1=\tan^{-1}\Delta$　である．

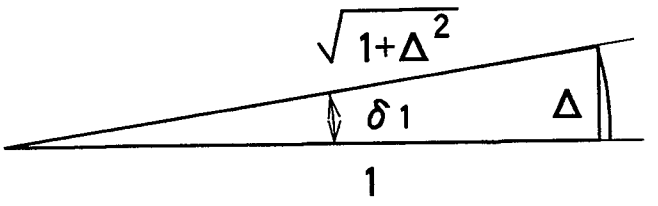


Fig.a.8　Relation between δ_1 and Δ

（２．３）ＦＩＨＴ２式

穂坂の与えた結果を引用するので省略．詳しくは文献(63)を参照のこと

（２．４）高精度漸化式

与えられた漸化式は

$$\rho_n=\begin{bmatrix}\sqrt{1-\Delta^2} & \Delta\\ -\Delta & \sqrt{1-\Delta^2}\end{bmatrix}\cdot\rho_{n-1}=\mathbf{A}\cdot\rho_{n-1}\quad \text{但し}\mathbf{A}=\begin{bmatrix}\sqrt{1-\Delta^2} & \Delta\\ -\Delta & \sqrt{1-\Delta^2}\end{bmatrix}\equiv\mathbf{A}(\Delta)$$

ここで新しい変数δ_2を用意し図a.9の如くΔとの関係を定めれば

$$\cos\delta_2=\sqrt{1-\Delta^2}\qquad \sin\delta_2=\Delta\qquad \delta_2=\tan^{-1}(\Delta/\sqrt{1-\Delta^2})$$

従って$\mathbf{A}(\Delta)$は

$$\mathbf{A}(\Delta)=\begin{bmatrix}\cos\delta_2 & \sin\delta_2\\ -\sin\delta_2 & \cos\delta_2\end{bmatrix}=\mathbf{A}(\delta_2)$$

ここで式(a.5)を用いて

$$\rho_n=\mathbf{A}(\Delta)\cdot\rho_{n-1}=\mathbf{A}(\Delta)^n\cdot\rho_0$$

$$=\begin{bmatrix}\cos\delta_2 & \sin\delta_2\\ -\sin\delta_2 & \cos\delta_2\end{bmatrix}^n\cdot\rho_0$$

$$=\begin{bmatrix}\cos(n\cdot\delta_2) & \sin(n\cdot\delta_2)\\ -\sin(n\cdot\delta_2) & \cos(n\cdot\delta_2)\end{bmatrix}\cdot\rho_0$$

これを展開して初期値$(\rho_{1,0},\rho_{2,0})^T=(x,y)^T$を与えれば

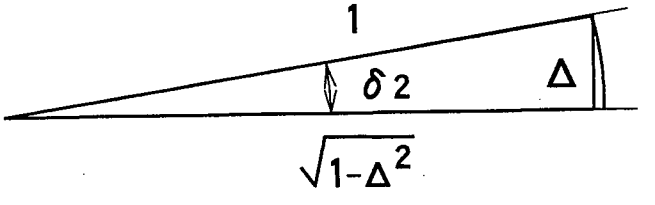


Fig.a.9　Relation between δ_2 and Δ

$$\rho_{1,n}=x\cdot\cos(n\cdot\delta_2)+y\cdot\sin(n\cdot\delta_2)$$

$$\rho_{2,n}=-x\cdot\sin(n\cdot\delta_2)+y\cdot\cos(n\cdot\delta_2)$$

となり式(8.36),(8.37)が得られた．ここで　$\delta_2=\tan^{-1}(\Delta/\sqrt{1-\Delta^2})$である．

（２．５）高速高精度漸化式

与えられた漸化式は

$$\rho_n=\begin{bmatrix}1-\Delta^2/2 & \Delta\\ -\Delta & 1-\Delta^2/2\end{bmatrix}\cdot\rho_{n-1}=\mathbf{A}\cdot\rho_{n-1}\quad \text{但し}\mathbf{A}=\begin{bmatrix}1-\Delta^2/2 & \Delta\\ -\Delta & 1-\Delta^2/2\end{bmatrix}\equiv\mathbf{A}(\Delta)$$

ここで新しい変数δ_3を用意し図a.10の如くΔとの関係を定めれば

$$\cos\delta_3 = \frac{1-\Delta^2/2}{\sqrt{1+\Delta^4/4}} \qquad \sin\delta_3 = \frac{\Delta}{\sqrt{1+\Delta^4/4}} \qquad \delta_3 = \tan^{-1}\{\Delta/(1-\Delta^2/2)\}$$

従って$\mathbf{A}(\Delta)$は

$$\mathbf{A}(\Delta) = \sqrt{1+\Delta^4/4}\cdot\begin{bmatrix}\cos\delta_3 & \sin\delta_3\\ -\sin\delta_3 & \cos\delta_3\end{bmatrix} = \sqrt{1+\Delta^4/4}\cdot\mathbf{A}(\delta_3)$$

但し

$$\mathbf{A}(\delta_3) = \begin{bmatrix}\cos\delta_3 & \sin\delta_3\\ -\sin\delta_3 & \cos\delta_3\end{bmatrix}$$

ここで式(a.5)を用いて

$$\boldsymbol{\rho}_n = \mathbf{A}(\Delta)\cdot\boldsymbol{\rho}_{n-1} = \mathbf{A}(\Delta)^n\cdot\boldsymbol{\rho}_0 = \{\sqrt{1+\Delta^4/4}\cdot\mathbf{A}(\delta_3)\}^n\cdot\boldsymbol{\rho}_0$$

$$= (\sqrt{1+\Delta^4/4})^n\cdot\begin{bmatrix}\cos\delta_3 & \sin\delta_3\\ -\sin\delta_3 & \cos\delta_3\end{bmatrix}^n\cdot\boldsymbol{\rho}_0$$

$$= (\sqrt{1+\Delta^4/4})^n\cdot\begin{bmatrix}\cos(n\cdot\delta_3) & \sin(n\cdot\delta_3)\\ -\sin(n\cdot\delta_3) & \cos(n\cdot\delta_3)\end{bmatrix}\cdot\boldsymbol{\rho}_0$$

これを展開して初期値$(\rho_{1,0}, \rho_{2,0})^T = (x, y)^T$を与えれば

$$\rho_{1,n} = (\sqrt{1+\Delta^4/4})^n\cdot\{x\cdot\cos(n\cdot\delta_3) + y\cdot\sin(n\cdot\delta_3)\}$$

$$\rho_{2,n} = (\sqrt{1+\Delta^4/4})^n\cdot\{-x\cdot\sin(n\cdot\delta_3) + y\cdot\cos(n\cdot\delta_3)\}$$

となり式(8.39),(8.40)が得られた. ここで $\delta_3 = \tan^{-1}\{\Delta/(1-\Delta^2/2)\}$である.

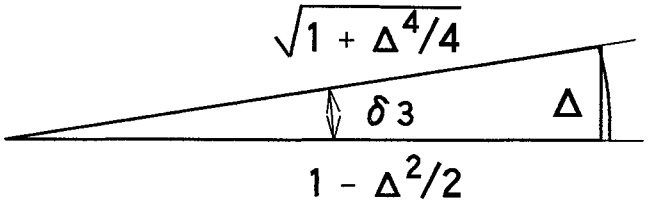


Fig.a.10 Relation between δ_3 and Δ

参考文献

(1) 村上，輿水，長谷川：Hough変換法のパターン形状特徴抽出法への一般化，電子情報通信学会技術報告，PRU88-3，(1987)，pp. 17-24

(2) A. Rosenfeld and A.C.Kak: Digital Picture Processing, Vol.2, ACADEMIC PRESS(1976), pp.251-25

(3) K.Inoue, K.Kimura: A Method for Calculating the Perimeter of Objects for Automatic Recognition of Circular Defects, NDT internatio-nal,Vol.20 No.4(1987),pp.225-230

(4) 井上，木村：「３画素ベクトル法」の原理について，溶接学会論文集，Vol. 5 No. 2(1987)，pp. 229-233.

(5) 井上，木村：「３画素ベクトル法」の応用，溶接学会論文集，Vol. 6 No. 2(1988)，pp. 251-255.

(6) 中島，大淵，井上：ＤＬＳ法による周囲長計測，溶接学会軽構造接合加工研究委員会資料，MP-89(1989)

(7) 中島，大淵，井上：隣接２線走査法の原理と縦長画素の採用による精度向上 -隣接２線走査法による周囲長計測（第１報）-，溶接学会論文集，Vol. 8 No. 4(1990)，pp. 470-475.

(8) 中島，大淵，井上：隣接２線走査法のハードウエア化とその性能評価 -隣接２線走査法による周囲長計測（第２報）-，溶接学会論文集，Vol. 9 No. 2(1991)，pp. 470-475.

(9) 中島，大淵，井上：直線及び円弧の２値化の過程，久留米高専研究紀要，Vol. 8 No. 1(1992)，pp. 15-20

(10) 中島，大淵，井上：スキップ２線走査を用いたディジタル測長法，電気学会論文誌Ｄ編，Vol. 113-D No. 3(1993)，pp. 357-363

(11) 中島，大淵，井上：スキップ２線走査を用いたディジタル測長法における部分長の生成と測長値：久留米高専研究紀要，Vol. 8 No. 2(1992)，pp. 33-39

(12) K.Nakashima, Y.Obuchi, K.Inoue: Betrachtung ü ber die Kontur Formgebung in Bin ä ren Bilder Zeugang, Transaction of JWRI., Vol.21 No.2(1992),pp.61-66

(13) K.Nakashima, Y.Obuchi, K.Inoue: Algorithm of Dual Lines Scanning Method and Accuracy Improvement by Employing Rectangular Pixel, Transactions of JWS.,Vol.24 No.1(1993),pp.39-44

(14) K.Nakashima, Y.Obuchi, K.Inoue:Implementation in Hardware and Its Evaluation of Dual Lines Scanning Method, Transactions of JWS.,Vol.24 No.1(1993),pp.45-49

(15) 中島，大淵，井上：表参照方式 Hough 曲線描画における表構成の効率化と高速化，テレビジョン学会論文誌”画像情報工学と放送技術”,Vol.48 No.6(1994),pp.695-701

(16) 中島，大淵，井上：ＣＣＤイメージセンサを利用した高速 Hough 変換，電子情報通信学会論文誌Ｄ－Ⅱ，Vol.J77-D-Ⅱ No.4(1994),pp.737-743

(17) K.Nakashima, Y.Obuchi, K.Inoue: High-Speed Hough Transform Employing CCD Area Image Sensor, Proc. of ACCV'93,Sec. 3C1 No.77(1993),pp.668-671

(18) 中島，大淵，井上：分割縮小化表参照方式による高速 Hough 変換ハードウエア，電

気学会論文誌D， Vol.115-d, No.5,(MAY 1995),pp.591
(19) D.BEN-TZVI and M.Sandler: ANALOGUE IMPLEMENTATION OF HOUGH TRANSFORM, ELECTRONICS LETTERS, Vol.25 No.18(1989),pp.1216-1217
(20) D.BEN-TZVI, A.A.Naqvi and M.Sandler: Efficient parallel implementation of the Hough Transform on a distributed memory system, Image and vision computing, Vol.7 No.3(1989),pp.169-172
(21) 輿水，沼田 ：区分的Hough直線による高速Hough変換法PLHTについて，信学論(D-Ⅱ), Vol.J72-7-Ⅱ No.1(1989),pp.56-65
(22) 花原，丸山，内山：実時間Hough変換プロセッサとそのFAへの応用，第3回日本ロボット学会講演会,No.2108(1985)
(23) 沼田，輿水：インクレメンタルな高速Hough変換法FIHT", 電子情報通信学会技術報告,PRU87-93(1988),pp.1-6
(24) 沼田，輿水：高速インクレメンタルHough変換法(FIHT2), 電子情報通信学会技術報告,PRU88-107(1989),pp.7-14
(25) 塩野:Hough変換の計算における3種類の表参照方式の効率比較実験，電子情報通信学会論文誌(D-Ⅱ)， Vol. J72-D-Ⅱ, No.6(1989),pp.963-966
(26) 安居院，亨，中嶋：ピラミッド階層化を利用した高速なハフ変換，テレビジョン学会技術報告， Vol.11 No.9(1987),pp.79-84
(27) 安居院，亨，中嶋：ピラミッド階層化高速Hough変換法を用いたナンバープレート領域抽出，電子情報通信学会論文誌(D),Vol.J70-D,No.7(1987),pp.1383-1389
(28) 沼田，輿水：ソフトウエアによるHough変換の高速実行，電子情報通信学会論文誌(D-Ⅱ),Vol.J73-D-Ⅱ,No.1(1990),pp.927-930
(29) Larry Werth: Automated Vision Sensing in Electronic Hardware, Sensors, (1986)
(30) 大橋，大和，石井，牧野：一般化Hough変換による任意図形検出アルゴリズム，電子情報通信学会技術報告,PRU88-123(1989),pp.33-39
(31) 安居院，中嶋：ピラミッド階層を利用した高速Hough変換について，電子情報通信学会技術報告,IE86-67(1987)
(32) 輿水：直線ﾊﾟﾀｰﾝ検出のためのHough追跡形アルゴリズムについて，電子情報通信学会論文誌 D,Vol.J69D No.4(1987),pp.631-633
(33) 沼田，輿水： Gradient型超高速Hough変換アルゴリズム，コンピュータビジョン，Vol.51 No.2(1987),pp.1-8
(34) 恩田，松島，青木： ROMを用いたHough変換ハードウエア，昭62電子情報通信学会全国大会,6-265(1987)
(35) 恩田，青木：三角関数の周期性を利用したHough変換の高速計算法，電子情報通信学会論文誌,Vol.70-D No.10(1987),pp.2009-2011
(36) Louis J. Galbiati, Jr: Machine vision and Digital Image Processing Fundamentals, Prentice Hall, (1990)
(37) Stephen B. Gray: Local Properties of Binary Images in Two Dimensions, IEEE Trans. on Computers, Vol.C-20 No.5(1971)
(38) William Green: Digital Image Processing, Van Nostrand Reinhold,(1982)
(39) C.P.Hu,Lin and C.F.Liao: Optical holographic Hough processor for machine vision, Proceeding of

the SPIE, Vol. 1319,(1990), pp.686
(40) D.Ben-Tzvi and M.B.Sandler: A combinational Hough transform, Pattern Recognition Letters. Vol.11 (March 1990),pp.167-174.
(41) T.H.Hou,L,Lin and P.D.Scott: A neural network-based automated inspection, System with an application to surface mount device, International Journal of Production Research, Vol.31, No5(1993), pp.1171-1187
(42) M.F.X.B.van Swaij,F.V.M.Catthoor and H.J.de Man: Driving ASIC architectures for the Hough transform, Parallel Computing, Vol. 16(1990),pp.113-121
(43) L.da Fontoura Costa and M.B.Sandler: The Binary Hough transform and its implementation, Proceedings of the SPIE, Vol.1251(1990),pp.183-193
(44) T.Hnif and M.B.Sandler:A contour-based Hough transform system, Michroprocessors and Microsystems, Vol. 18, No.1(JAN 1994),pp. 19-26
(45) G.L.Dempsey and E.S.Mcvey: A Hough transform system based on neural networks, IEEE Transaction on Industrial Electronics, Vol..39, No.6(Dec 1992),pp.522-528
(46) 中島，大淵，井上：高速 Hough 変換のための漸化式とその一般解の導出及びインプリメンテーション，高温学会誌， Vol.22 No.1(1995),pp.48-56
(47) 中島，大淵，井上：連立漸化式による高速，高精度 Hough 変換法，信学論 D-Ⅱ，Vol.J79-D-Ⅱ,No.9(1996),pp.1509-1515
(48) 中島，大淵，井上：並列型連立漸化式による Hough 変換の高速実行，久留米高専研究紀要，投稿中
(49) V.F.Leavers: Shape Detection in Computer Vision Using the Hough Transform, Appendix 4, Springer-Verlag(1992)pp.184 ～ 185
(50) マイクロソフト(株):「コンピュータ用語辞典」，アスキー出版(1974),pp.302
(51)L.G.Roberts; Machine perception of three dimensional solids in Optical Processing of Information, MIT Press,1965,pp.159-197
(52) 輿水： Hough 変換に関する最近の研究動向，情処学研資，CV51-1(1987)， pp.1-8
(53) Hough. p. v.c: Method and means for recognizing complex patterns, U.S.Patent 3069654, (1962)
(54) R. O. Duda and P. E. Hart: Use of the Hough Transformation to Detect Lines and Curves in Pictures, Comm.ACM, Vol.15, No.1(Jan. 1972)
(55) 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，Vol.J69-D,No.4(Apr. 1986),pp.631-633
(56) 松山，長尾： Hough 変換の幾何学的性質と直線群検出への応用，情処学論誌，Vol.26,No.6(Jun.1985)， pp.1069-1078
(57) Wahl.F.M. and Biland.H.P.: Decomposition of poruhedra scene in Hough space, Proc.8th-ICPR, (Oct.1986),pp.78-84
(58) 村上，輿水，長谷川：凸包の高速計算アルゴリズム，信学技報， IE86-122 (March 1987)
(59) 村上，輿水，長谷川： Hough 変換平面における図形の凸包抽出アルゴリズムについて，情報処理学会研究報告，CV51-3(Nov.1987)
(60) 小島，木村，小沢：線分構造抽出のための新しい手法の提案，信学技報， Vol.88

No.324(1988), pp.5-8
(61) 大和, 稲葉, 石井, 牧野 : Hough変換を用いた線分検出の高精度化, 信学論, Vol.J72-D- Ⅱ ,No.1(1989),pp.89-92
(62) 松山、長尾 : Hough変換の幾何学的性質と直線群検出への応用, 情報学論, Vol.26,No.6,(1985),pp.1069-1078
(63) 穂坂 : コンピュータグラフィックス, 産業図書 (1974),pp.84-86
(64) R.ツルミュール : マトリックスの理論と応用, ブレイン図書出版 (1972)
(65) 鎌田 : Hough変換を応用した無人走行システム, SICE'95, 第32回パターン計測部会資料 (Jun 1995),pp.31-38
(66) 中島, 大淵, 井上 : 高速Hough変換と実時間形状計測を応用した矩形物体の検査, テレビジョン学会論文誌"画像情報工学と放送技術" ,Vol.50 No.1 (1996),pp.695-701
(67) 塩野 : パラメータ平面を使用しない高精度Hough変換を用いた近接2直線の識別, 信学技報, Vol.88,No.450(1989),pp.41-48
(68)加藤, 山崎, 遠藤, 村上, 鳥生, 輿水 : エッジ点のランダムな投票によるHough変換に関する考察, 電気学会論文誌C, Vol.117-C No.1(1997),pp.81-86

本研究に関する発表論文

(1) 隣接2線走査法の原理と縦長画素の採用による精度向上 -隣接2線走査法による周囲長計測（第1報）-,溶接学会論文集, Vol.8 No.4(1990),pp.34-39.
(2) 隣接2線走査法のハードウエア化とその性能評価 -隣接2線走査法による周囲長計測（第2報）-, 溶接学会論文集, Vol.9 No.2(1991),pp.20-25.
(3) Betrachtung über die Kontur Formgebung in Binären Bilder Zeugang,Trans.JWRI,Vol.21 No.2(1993),pp.61-66
(4) Algorithm of Dual Lines Scanning Method and Accuracy Improvement by Employing Rectangular Pixel,Trans.JWS,Vol.24 No.1(1993),pp. 39-44
(5) Implementation in Hardware and Its Evaluation of Dual Lines Scanning Method,Trans.JWS,Vol.24 No.1(1993),pp. 45-49
(6) 直線及び円弧の2値化の過程, 久留米高専研究紀要, Vol.8 No.1(1992),pp.15-20
(7) スキップ2線走査を用いたディジタル測長法, 電気学会論文誌D編, Vol.113-D No.3(1993),pp.357-364
(8) スキップ2線走査を用いたディジタル測長法における部分長の生成と測長値：久留米高専研究紀要, Vol.8 No.2(1992),pp.33-39
(9) 表参照方式 Hough 曲線描画における表構成の効率化と高速化,テレビジョン学会論文誌"画像情報工学と放送技術",Vol.48 No.6(1994),pp.695-701
(10) CCDイメージセンサを利用した高速 Hough 変換, 電子情報通信学会論文誌D-II, VolJ77-D- II No.4(1994),pp.737-743
(11) High-Speed Hough Transform Employing CCD Area Image Sensor, Proc. of ACCV'93,Sec. 3C1 No.77(1993),pp.668-671
(12) 分割縮小化表参照方式による高速 Hough 変換ハードウエア, 電気学会論文誌D, Vol.115-D,No.5,(May 1995),pp.591-597
(13) 高速 Hough 変換のための漸化式とその一般解の導出及びインプリメンテーション, 高温学会誌, Vol.22,No.1(1995),pp.48-56
(14) 高速 Hough 変換と実時間形状計測を応用した矩形物体の検査, テレビジョン学会論文誌"画像情報工学と放送技術",Vol.50,No.1(1996),pp.103-110
(15) 連立漸化式による高速, 高精度 Hough 変換法, 信学論D-II ,VolJ79-D- II No.9,(Sep.1996),pp.1509-1515
(16) ハフ変換の漸化式導出と多元化による高速計算, 久留米高専研究紀要, Vol. 12, No. 2 (Mar. 1997), pp. 1-3
(17) High Speed, High Accuracy Hough Transform Using Simultaneous Recurrence Formula,Systems and Computers In Japan,(Apr.1997), John Wiley and Sons Inc.

本研究に関する特許

(1)度数累算型メモリ半導体集積回路 特許 第303477号
(2)ハフ変換演算装置 出願中 特願平8-121695号