ディジタル直線検出問題の計算量に関するアルゴリズム論的考察

浅野　哲夫†　　河村　泰之†

Algorithmic Considerations on the Computational Complexities of Digital Line Extraction Problem

Tetsuo ASANO† and Yasuyuki KAWAMURA†

あらまし　白黒 2 値画像に含まれる直線成分を検出する方法として，コンピュータビジョンの分野では Hough 変換が使われることが多い．パラメータ空間における投票という概念のわかりやすさが好んで利用される最大の理由であろう．本論文では，与えられた画像に含まれる直線成分をすべて検出することという制約のもとで，このような投票に基づく方法の計算量の下界について議論する．最後に，計算幾何学で開発された技法に基づいて，全く異なる方法を提案し，その計算量，作業領域の面での優位性を論じる．

キーワード　計算量，コンピュータビジョン，双対変換，直線検出，ディジタル直線，Hough 変換

1. まえがき

与えられた 2 値画像に含まれるすべての直線成分を抽出する問題はパターン認識における基本的な問題の一つであり，Hough 変換 [12], [13], [19] の名前で知られる多くの研究がある [1], [10], [11], [14], [16]〜[18], [20]〜[24]．その大部分は，問題を細分化されたパラメータ空間に写像して，その上で投票を行うという考え方に基づくものである．

基本的な考え方は次のとおりである．エッジ点を通る直線を，その法線角度 θ と原点から直線までの距離 ρ を用いてパラメータ化したとき，エッジ点は $\rho\theta$-パラメータ平面における曲線となるが，そのような曲線同士の交点は，対応するエッジ点を通る直線に対応する．したがって，多数の曲線が交差する場所を求めれば直線検出ができることになる．実際には，エッジ点が整数座標をもつので，多数のエッジ点が正確に一直線上にあることは少ないので，一定の幅をもたせて考える必要がある．そこでパラメータ平面を小領域に区切って，一つの小領域を通過する曲線が一定数以上あれば，その小領域に対応する直線を報告することにより，上記の幅を考慮したことにするというものである．

同様の性質をもつ変換として，点と直線の間の双対変換がある．これは，点を定める二つのパラメータ（x, y 座標値）を直線を定める二つのパラメータ（傾きと y 切片）に対応させるものであり，双対平面における多数の直線の交点は元の平面において多数の点を通る直線に対応する．両者を比較したとき，Hough 変換の場合には，パラメータ平面を $0 \leqq \theta \leqq \pi$, $0 \leqq \rho \leqq \sqrt{2}N$ の範囲に制限できるのに対して，双対変換の場合には傾きの範囲を限定できないという事情がある．ただし，N は画像の長辺の長さ（画素数）である．

パラメータ平面を小領域に区切っておき，曲線と交差をもつ小領域に印を付けていくというアルゴリズムの技法が投票と呼ばれているものである．この方法はプログラム化が容易であるのと，比較的良好な結果が得られるので普及しているが，理論的には残された問題点は多い．例えば，パラメータ平面の小領域への分割は最大の問題であるが，分割が粗いと直線検出能力の面で劣り，分割を細かくすると計算時間がかかるというトレードオフがあり，最適な分割を定義することが難しい．最も素朴な分割方法は，パラメータ平面を水平垂直線で一様に分割するというものであるが，この方法では角度によって直線抽出能力に顕著な差が生じることが指摘されており，そのようなパラメータ平面のひずみを除去する方法が提案されている [24]．

本論文では，検出すべき直線成分を数学的に定義した上で，定義の条件に合致するすべての直線成分を検出するという前提で，投票に基づく方法の計算量的限

† 北陸先端科学技術大学院大学情報科学研究科，石川県
School of Information Science, Japan Advanced Institute of Science and Technology, Ishikawa-ken, 923–1292 Japan

界を示す．また，従来の方法では各角度について対応する曲線の位置を計算し，丸めの計算を行うことによって 1 個の投票箱を決定する，すなわち，各角度について 1 票を投じるという方式が一般的であるが，すべての直線成分を検出するためには，各角度に対して多数の投票を行わなければならないことを示す．これが非効率の主たる原因である．最後に，投票という技法を全く用いない新たな手法を提案し，その有効性，優位性を考察する．

2. 標準的な Hough 変換

コンピュータビジョンの分野では，エッジ検出後の 2 値画像から直線成分を検出するのに Hough 変換と呼ばれる手法が用いられることが多い．これは，元来，1960 年代初頭に Hough によって提案された方法を指すが，それ以来様々な改良が加えられてきている．特に，Duda と Hart による改良案が広く支持され，この改良案が一般に Hough 変換と呼ばれている．まず，この方法を説明し，計算量を解析するとともに，どこに問題点があるかを明確にしたい．

画像平面上のエッジ点（エッジがあると判断された点–あるいは，黒点）の座標を (x_i, y_i)，そのエッジ点を通る直線と原点の距離を ρ，その直線の法線が x 軸となす角を θ とするとき，この直線は

$$\rho = x_i \cos\theta + y_i \sin\theta$$

と表現できる（図 1 参照）．Hough 変換では，上式を xy-座標の点 (x_i, y_i) から $\rho\theta$-パラメータ平面上の曲線 $\rho = x_i \cos\theta + y_i \sin\theta$ への変換とみなす．重要な性質は，xy-平面上の 2 点を通る直線を定めるパラメータ値 ρ, θ は，それらの 2 点に対応する曲線の交点の座標として与えられるということである．この性質より，画像平面における直線成分を求める問題を，パラメータ平面において多数の曲線が交差する点を求める問題に変換することができる．

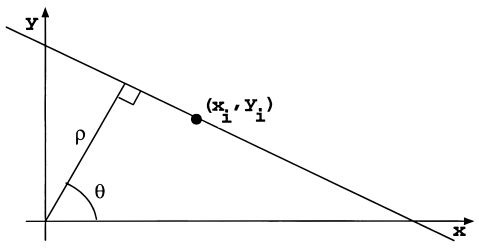

図 1　Hough 変換の原理
Fig. 1　The principle of the Hough Transform.

多数の点が正確に一直線上に存在すれば，それらに対応する曲線は正確に 1 点で交差するが，画像平面上の点は整数座標であるために曲線の交点は多少ずれる．このずれを吸収するために，投票という方法が用いられる．すなわち，パラメータ平面を小さなセル（長方形の領域）に分割することによって離散化する．角度 θ は区間 $[0, \pi]$ に限定でき，距離 ρ も区間 $[0, \sqrt{2}N]$ に制限することができる．ただし，N は画像の 1 辺の長さである．

通常は，このように制限されたパラメータ平面を $(N \times N)$ のサイズのセルに分割しておく．各エッジ点 (x_i, y_i) に対して，角度 θ を変化させながら，対応する曲線 $\rho = x_i \cos\theta + y_i \sin\theta$ を順にたどり，この曲線が通過するセルに投票を行っていく．すべてのエッジ点について投票を終えたとき，一定数以上の得票を得たセルを列挙し，それらのセルに対応する直線を出力する．

以上が処理の概略である．厳密には，曲線が通過するセルをすべて列挙するのは計算に時間がかかるため，角度を $O(N)$ 通りに量子化し，それぞれの角度において対応する ρ の値を計算し，その値を更に量子化することによって投票すべきセルを指定するのが一般的である．したがって，各量子化された角度について 1 回だけ投票を行うことになる．

したがって，アルゴリズムは以下のようになる．

［アルゴリズム 1：標準的な Hough 変換］
用意すべき配列：
（1） bit $g[N][N]$; /* 入力 2 値画像 */
（2） int vote$[M_\theta][M_\rho]$; /* 投票箱 */
パラメータ：
（1） $(\theta_1, \theta_2, \ldots, \theta_{M_\theta})$: /* 角度の系列 */
（2） ρ の値を $1 \sim M_\rho$ に量子化する関数 $q(\theta, \rho)$.
（3） 直線成分として認めるために必要なエッジ点の個数に関するしきい値 t.

投票箱 vote[.][.] を 0 に初期化；
for $x = 0$ to $N - 1$ do
for $y = 0$ to $N - 1$ do
　if 画素 (x, y) がエッジ点 then
　　for all 角度 $\theta_i \in [1, M_\theta]$ {
　　$\rho = x \cos\theta_i + y \sin\theta_i$ を計算；
　　投票箱 vote$[i][q[\theta_i, \rho]]$ に 1 票入れる；
　　}

```
for all 角度 θ_i ∈ [1, M_θ] do {
  for all 距離 ρ_j do
    if 得票数 vote[i][j] ≧ t then
      (θ_i, ρ_ij) に対して，対応する直線の式
        x cos θ_i + y sin θ_i = ρ_ij
      を出力；
}
```

3. 標準的な Hough 変換の問題点

上に述べた Hough 変換法は形式的なものであり，そのままの形では多くの問題点を含んでいる．整理すると以下のようになる．

（1） しきい値の選択：直線成分を構成するために必要な最低限度のエッジ点数を定めたしきい値 t をどのように選ぶかは重要な問題であるが，妥当なしきい値は画像のサイズや種類によっても異なり，また，画像全体で同じしきい値を適用して良い結果が得られるという保証もない．

（2） 極大直線成分：少しでも冗長な直線成分の出力を避けるため，しきい値以上の得票を得たセルをすべて出力するのではなく，得票数が局所的に最大であるものだけに限定するのが一般的である．ただし，後で述べるように，このような制約が妥当かどうかはパラメータ平面の分割に強く依存しており，経験的な知識が要求される曖昧（あいまい）な世界である．

（3） パラメータ平面のひずみ：パラメータ平面のひずみに関しては多数の報告がある（例えば，[24]）．パラメータ平面の分割方法として最も単純な方法は，角度と距離の区間を均等に分割するというものであるが，均等分割では直線検出の性能が良くないことが知られている．更に，セル分割と検出される直線成分の性質との関連も曖昧である．すなわち，セルのサイズと，対応する直線成分の角度と距離の幅とはどのような関係にあるのかが不明確である．

（4） 端点情報：上記のアルゴリズムでは，得票数だけを判断基準にして直線成分を抽出しているので，基本的に端点をもつ線分の検出は難しい．

（5） 密度制約：上記と同様の理由により，同じ個数の点が一直線上にあれば同じ得票数となるので，画像の端から端までまばらに点が存在する場合と，ある狭い範囲で点が連続している場合の区別がつかない．実際にはある密度以上で点が連続している場合にだけ直線として認識すべきであろうが，そのような密度制約を投票操作にどのように組み入れるべきかは知られていない．

4. 直線成分の定義と特徴付け

まず，抽出すべき直線成分を次のように定義し，定義に基づいて直線成分の特徴付けを行う．

［直線成分の定義］

直線 $y = ax + b$ をディジタル画面上に表現するのによく使われるのは，この直線までの垂直距離または水平距離が 0.5 以内であるような格子点の集合を用いるものである．そのような格子点の集合を直線 $y = ax + b$ のディジタル表現，あるいはディジタル直線といい，$G(a, b)$ と表す．このとき，あるエッジ点の集合 P に対して，$P \subseteqq G(a, b)$ となるような a, b が存在するとき，この集合 P は（ディジタル）直線成分をなすという．極大（ディジタル）直線成分とは，他のどのエッジ点を加えてもディジタル直線成分の条件を満たさなくなるものをいう．

直線の傾きの絶対値が 1 より大きいときは，水平距離の方が垂直距離より必ず小さく，逆に直線の傾きの絶対値が 1 より小さいときは，垂直距離の方が水平距離より必ず小さい．本論文では傾きの絶対値が 1 以下の直線だけを対象にするが，縦軸と横軸を入れ替えれば同じ議論が成り立つことを考慮すると，この制限によって一般性を失うことはない．

さて，このように直線成分を定義したとき，直線成分抽出問題は次のように記述できる．

［直線成分抽出問題］

n 個のエッジ点を含む $N \times N$ の 2 値画像と，しきい値 t が与えられたとき，この画像に含まれるサイズが t 以上の極大ディジタル直線成分をすべて列挙せよ．

ここでの目標を列挙すると次のようになる．

目標 1　極大でないものを出力せずに，極大ディジタル直線成分をすべて列挙することは可能か．

目標 2　計算誤差による誤出力がないこと．

目標 3　作業用の記憶スペースが妥当であること．

目標 4　計算時間も十分に短いこと．

最初の目標以外はいずれも曖昧な表現になっているが，本論文で目指すところは実験的な検証ではなく，理論的解析を与えることである．

ディジタル直線を上記のように定義したとき，逆にある格子点の集合 P がディジタル直線になっている

とすると，P の各点 $p_i(x_i, y_i)$ について，

$$-\frac{1}{2} \leqq y_i - ax_i - b \leqq \frac{1}{2}$$

を満たす実数 a, b が存在する（注意：傾きの絶対値が 1 以下のものだけを対象にしているので，垂直距離だけを考えればよい）．

さて，パラメータ対 (a, b) に対応するディジタル直線を

$$G(a,b) = \left\{ (x_i, y_i) \in G \middle| -\frac{1}{2} \leqq y_i - ax_i - b \leqq \frac{1}{2} \right\}$$

として定義したが，同じディジタル直線を与えるパラメータ対は等価であるとして，すなわち，

$(a, b) \equiv (a', b') \Longleftrightarrow G(a, b) = G(a', b')$

として，関係 $\equiv$ を定義するとき，これは明らかに同値関係であり，パラメータ平面を同値類に分割する．パラメータ平面は実数値で定義されるものであるが，どの同値類も必ず有理点（a, b の値が共に有理数であるもの）を含んでいることを示そう．

［補題 1］ 任意のディジタル直線成分 P に対して，P を特徴づけるパラメータ値の組 (a, b)，すなわち，$P \subseteqq G(a, b)$ となる (a, b) の中で，a が $\frac{q}{p}$（ただし，$0 \leqq q \leqq p \leqq N-1$）の形の有理数であり，$b$ が $b = y - \frac{q}{p}x \pm \frac{1}{2}$（ただし，$0 \leqq x, y \leqq N-1$）の形の有理数であるようなものが必ず存在する．

（証明） 定義より，P のすべての点 $p_i(x_i, y_i)$ に対して，

$$-\frac{1}{2} \leqq y_i - ax_i - b \leqq \frac{1}{2}$$

を満たす実数 a, b が存在し，$P \subseteqq G(a, b)$ が成り立つ．このエッジ点集合は，2 本の平行直線 $y = ax - b - \frac{1}{2}$ と $y = ax - b + \frac{1}{2}$ でに囲まれるベルト領域（の内部または境界上）に含まれるエッジ点の集合に等しいが，含まれるエッジ点の集合を変えないように 2 本の平行直線を下方に（すなわち，b の値を減らすように）平行移動して，上部または下部の境界線が整数格子点に接触するようになったところで止める．

次に，境界上の整数格子点を中心に上下 2 本の境界線を反時計回りに回転させ（すなわち，a の値を増加させて），上部または下部の境界が別の整数格子点に接触したところで止める．

上記の操作が完了したとき，二つの整数格子点 p_j と p_k が上部または下部の境界線上にあり，しかも 2 本の境界線の垂直方向の差は正確に 1 である．したがって，どちらの境界線も少なくとも二つの格子点（P のエッジ点とは限らない）を通ることになる．そのような 2 点の x 座標の差 p は，$-N+1 \leqq p \leqq N-1$ の範囲にあり，y 座標の差 q は，$p = 0$ でない限り，$\frac{q}{p}$, $|p| \leqq N-1, |q| \leqq N$ の形の有理数として表現することができる．したがって，$b = y - \frac{q}{p}x \pm \frac{1}{2}$ として与えられる b も有理数である． □

5. 投票に基づく方法とその限界

補題 1 より，傾き a は全部で $O(N^2)$ 通りしかないことがわかる．y 切片 b については，x と y がそれぞれ N 通りの値を取り得るから，全部で $O(N^4)$ 通りの値が考えられる．y 切片の値はたかだか $2N$ の整数を共通の分母とする分数として表現できるから，それらの値をソートするのは線形時間で十分である．

$O(N^2)$ 通りの傾きと $O(N^4)$ 通りの y 切片について投票箱を定義すると，投票箱の個数は $O(N^6)$ となってしまう．アルゴリズムを記述すると以下のようになるが，これは以前に与えたアルゴリズム 1 と本質的には同じものである．

［アルゴリズム 2］
投票箱をすべて空にしておく；
for all エッジ点 $p = (x, y)$ do
for all 傾き a_i do
　for all y 切片 b_j do
　　if $-\frac{1}{2} \leqq y - a_i x - b_j \leqq \frac{1}{2}$ then
　　　投票箱 $vote[i][j]$ に 1 票を入れる；
for all 傾き a_i do
for all y 切片 b_j do
　if 投票箱 $vote[i][j]$ の票数 $\geqq t$ then
　　直線 $y = a_i x + b_j$ を出力；

計算時間は，$O(N^6 + n \times N^2 \times N^4 + N^2 \times N^4) = O(nN^6)$ であり，必要な作業用スペースは $O(N^6)$ ということになる．これでは全く非実用的であるので，計算時間と作業用スペースの両面での改善が必要である．

上ではすべての y 切片について判定しているが，条件を満たす最小の b の値 b_j と最大の b の値 b_k を求めて，b_j から b_k までの値について投票を行うようにすると，若干の改善が可能である．

b_j と b_k は 2 分探索により $O(\log N)$ 時間で求めることができるが，j と k の値は $O(N^2)$ 程度離れることがあるので，各傾きに対して $O(N^2)$ 回

の投票を行うことになる．したがって，計算時間は，$O(N^6+n\times N^2\times(\log N+N^2)+N^2\times N^4)=O(N^6)$ となり，時間を短縮できる．ただし，$n=O(N^2)$ であることに注意．

上記のアルゴリズムは本質的には Hough 変換法と同じであるが，1 次式に基づいているため，計算誤差対策は容易である．また，先に述べた定義の条件を満たすディジタル直線成分をすべて列挙することができる点も従来の Hough 変換法との違いである．従来の Hough 変換法では，一つの傾きに対して 1 回しか投票を行わなかったが，上記の方法では $O(N^2)$ 回も投票を行っている．しかしながら，極大直線成分だけを出力するという点については悲観的である．上記の方法では極大性を確かめることは困難である．

すべての直線成分を列挙するという点では改善されたが，このままの計算時間では非実用的である．では，上記のアルゴリズムにおける無駄は何であろうか．

y 切片の値は全部で $O(N^4)$ 通りあり得るが，傾き a を固定すれば，x,y の値を変えるだけなので，全部で $O(N^2)$ 通りしかない．すなわち，各傾きに対して，それぞれ異なる y 切片の列を用いる．そのような列を生成するのは列の長さの線形時間，すなわち $O(N^2)$ 時間で可能である．このようにしたとき，以前のように b_j と b_k の幅は $O(N^2)$ ではなく，$O(N)$ ですむので，投票総数自身が減り，計算時間も削減できる．

［アルゴリズム 3］
for all 傾き a_i do {
　対応する y 切片の列 $(b_0,b_1,\ldots,b_{K_i-1})$ を求める；
　（ソートも含めて $O(K_i)=O(N^2)$ 時間でできる）
　投票箱 $vote[j], j=1,\ldots,K_i$ を空にしておく；
　for all エッジ点 $p=(x,y)$ do {
　　$-\frac{1}{2}\leqq y-a_ix-b_j$ を満たす最小の b_j を求める；
　　$y-a_ix-b_j\leqq\frac{1}{2}$ を満たす最大の b_k を求める；
　　for $h=j$ to k do
　　　投票箱 $vote[h]$ に 1 票を入れる；
　}
　for all y 切片 b_j do
　　if 投票箱 $vote[j]$ の票数 $\geq t$ then
　　　直線 $y=a_ix+b_j$ を出力；
}

上記のアルゴリズムではループで傾きとエッジ点を変化させる順序を変更していることに注意しよう．計算時間は，$O(N^2\times n\times(N^2+N^2+2\log N+N+N^2\times N^2)=O(nN^4)$ となり，改善は見られないが，特筆すべきことは，記憶スペースの減り方である．今までは 2 次元の表を必要としたが，今度は各傾きで投票が完結するので，1 次元の表で十分であり，全体で $O(N^2)$ で十分である．

更に，アルゴリズム 3 では直線成分の極大性をある程度確認できる．具体的には，投票箱の境界の直線上にエッジ点があるかどうかを判断すればよい．詳しくは述べないが，この判定を毎回行うと $O(N)$ 時間余分にかかるが，他の処理の中に埋め込むことも可能である．この点については後に述べるアルゴリズムの中で再度詳細に説明する．ただし，上である程度極大性が判定できるといったのは，同じ傾きで y 切片だけが違うときにのみ適用可能で，傾きが異なる直線成分は極大性判定の対象にしていないからである．

今度のアルゴリズムにおける無駄は，傾きを決めた後，投票箱の初期化に $O(N^2)$ 時間，投票そのものに $O(N)$ 時間，投票数の確認に $O(N^2)$ かかっていることである．これらの時間を削減したい．そのためにデータ構造を工夫する．

今まで，投票数は 1 次元の表で管理していた．1 回の投票で一つの配列要素にのみ投票するのであれば 1 次元の表で十分であるが，上記のアルゴリズムでは一つの傾きに対して，j 番目から k 番目までの（連続する）投票箱に投票を行っている．すなわち，区間で投票を行っているのである．このように考えると，区間を管理するデータ構造 [3] を用いるのが妥当であることがわかる．

各傾きに対して投票箱の個数は $O(N^2)$ に限定できるが，具体的な個数は一般に異なる．そこで，その最大値を M として，1 から M までの区間を扱う区分木を用いる．これは，次のように定義することができる．

全体区間 $[1,M]$ を次のようにして部分区間に分割し，各部分区間を節点に対応させることによって木を構成する．具体的には，まず，区間 $[1,M]$ をその中央の値 $m=\lfloor M/2\rfloor$ によって $[1,m]$ と $[m+1,M]$ に 2 分割する（記号 $\lfloor\ \rfloor$ は小数点以下の切捨てを意味する）．これらの部分区間は同様にして更に 2 分割され，これを区間の左端と右端が一致するまで続ける．このようにして作られる部分区間を節点に対応させ，区間 I を 2 分割して区間 I_1 と I_2 ができるとき，これらの部分区間に対応する節点を I に対応する節点の子節点とする．以下では，区間 $[u,v]$ に対応する節点を $T(u,v)$ として表すことにする．

このようにデータ構造を定めたとき，アルゴリズムの中で区間 $[j,k]$ に対する投票は次の手続き voting( ) を根に対して適用すればよい（具体的には voting(1, M,j,k) を呼び出す）．

手続き voting (u,v,j,k)
（1） 節点 $T(u,v)$ の区間 $[u,v]$ が投票区間 $[j,k]$ に完全に含まれるなら，節点 $T(u,v)$ の得票数に 1 を加えて終わり．
（2） 節点 $T(u,v)$ の左の子 $T(u,v')$ の区間と投票区間 $[j,k]$ が共通部分をもつなら，節点 $T(u,v')$ にも同じ操作 voting (u,v',j,k) を再帰的に適用する．
（3） 節点 $T(u,v)$ の右の子 $T(u',v)$ についても（2）と同様の操作を行う．

簡単な解析により，上記の手続きにおいて訪問される節点の個数は $O(\log N)$ であることがわかる．すべてのエッジ点に関する投票が終わると，投票結果の集計をする必要がある．これは，すべての葉節点について，自分自身を含めて，その先祖の節点の得票数の総計を求めることを意味している．したがって，根から葉に向けてそれぞれの得票数を子に足し込んでいけばよい．これは明らかに区分木のサイズ $O(N^2)$ で実行できる．

傾きを変えたとき，区分木のデータは初期化する必要があるが，これも $O(N^2)$ の時間で可能である．毎回の初期化を省略するためのトリックも考えられるが，本質的でない割に煩雑であるので，ここでは述べない．

結局，全体の計算時間は，$O(N^2\times(N^2+n\times\log N+N^2))=O(nN^2\log N+N^4)$ ということになり，先の計算時間 $O(nN^4)$ に比べると大幅な改善である．

上記をまとめると，アルゴリズムを本質的に変更しない限り，究極的に $O(nN^2\log N)$ より速くすることは不可能であるように思われる．

6. 投票に頼らない方法

上に述べたように，投票に頼る限り $O(nN^2\log N)$ 程度の計算時間は必要になるが，全く異なる考え方で $\log N$ のファクタを取り除くことができる．これは，筆者らが以前に発表したもので [5]，投票を全く使わないものである．前記のアルゴリズムと同様に傾きを順に変化させながら，直線成分を求めていく．このアルゴリズムでは，傾き a_i に対して，各エッジ点 $p_j(x_j,y_j)$ から原点を通る直線 $y=a_ix$ までの垂直距離 $v_j=y_j-a_ix_j$ を計算し，この値をキーとしてエッジ点をソートすることにより，垂直幅が 1 であるエッジ点の集合を直線成分として求める．具体的には以下のとおりである．

［アルゴリズム 4（Asano-Katoh '96 [5]）］
for all 傾き a_i do {
　for all エッジ点 $p_j=(x_j,y_j)$ do
　　$v_j=y_j-a_ix_j$ の値を計算；
　エッジ点を v_j の値の昇順にソート；
　ソート列を走査して，v_j の値の幅が 1 以内でサイズが t 以上の部分をすべて列挙し，それらに対応する直線を出力する；
}

傾きの系列を Farey 数列として生成するとき，上記のソート処理は線形時間で可能である．したがって，計算時間は $O(nN^2)$ である．また，作業用のスペースとしては，エッジ点の管理に $O(n)$，傾きの列を管理するのに $O(N^2)$ だけ必要であるから，全体では $O(n+N^2)$ となる．ただし，毎回傾きを Farey 数列の要領で生成することにすれば，傾きを配列に入れて管理する必要はないが，再帰呼出しの戻り番地を記憶するのに $O(N)$ 程度のスペースが必要である．

投票という操作は用いないものの，アルゴリズムとしてはほとんど同じ構造であるので，上記の投票に基づく方法と同様，異なる傾きの直線成分間では集合としての包含関係を調べて，極大でないものを排除することはできない．

7. アレンジメントに基づく方法

上に述べたのと全く異なる考え方に基づく方法として，パラメータ平面上における直線のアレンジメントに基づく方法がある．その基本的な考え方は文献 [5] でも示されているが，ここでは更に洗練されたアルゴリズムを示す．

基本的な考え方は次のとおりである．エッジ点 $p(x,y)$ がディジタル直線成分 $G(a,b)$ に含まれるのは，

$$-\frac{1}{2}\leqq y-ax-b\leqq\frac{1}{2}$$

が成り立つときである．この式を書き換えると，

$$-ax+y-\frac{1}{2}\leqq b\leqq-ax+y+\frac{1}{2}$$

となる．これを a-b 平面で考えると，2 本の平行直線で囲まれたベルト領域に対応している（図 2 参照）．

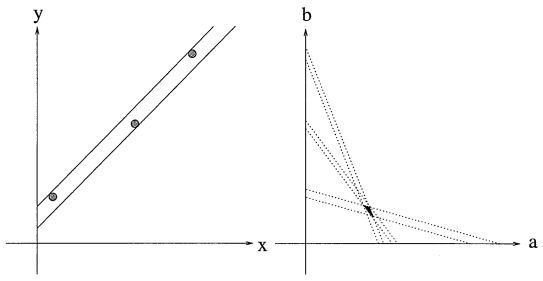

図 2　アレンジメントの原理
Fig. 2　Arrangement of lines.

つまり，各エッジ点が一つのベルト領域に対応することになる．二つのエッジ点のベルト領域が共通部分をもつとき，その共通部分の任意の点を (a, b) とすると，この二つのエッジ点は直線 $y = ax + b$ のディジタル直線成分に含まれることになる．したがって，各エッジ点について対応するベルト領域を求めた後，なるべく多数のベルト領域が交差する部分を求めればよいことになる．

各エッジ点に対応する $2n$ 本の直線の交差関係をすべて調べてグラフ構造を構成することにより，これらの直線群によって区切られる小領域（セルと呼ぶ）それぞれについて，それを含むベルト領域の集合を特徴づけることができる．これを素朴な方法で実行すると，$O(n^3)$ 時間と $O(n^2)$ の記憶スペースが必要になるが，平面走査法を用いると，記憶スペースを $O(n)$ に，計算時間を $O(n^2 \log n)$ に短縮することができる．更に，Asano-Guibas-Tokuyama [4] の Topological walk アルゴリズムを用いてアレンジメントの中を探索することにすると，記憶スペースは線形 $O(n)$ のままで，計算時間を $O(n^2)$ に削減できる．

さて，ベルト領域は平行な 2 本の直線で囲まれた領域であるが，上下の直線をそれぞれプラス辺，及びマイナス辺と呼ぶことにする．これらの直線によりパラメータ平面はセルと呼ばれる小多角形に分割されるが，それらの多角形はいずれも凸多角形である（図 2 参照）．したがって，セルの上部境界と下部境界が自然に定義できる．このとき，セル C に対応する直線成分が極大であるためには，上部境界はすべてプラス辺であり，下部境界はすべてマイナス辺でなければならない．例えば，セル C の上部境界にマイナス辺があったとすると，この辺を超えてすぐの点は，セル C に対応するエッジ点たちの他に，隣接するベルト領域に対応するエッジ点も含んだ直線成分に対応しており，極大性に反する．また，セル C の下部境界にプラス辺があったとすると，この辺を超えてすぐの点は，セル C に対応するエッジ点たちから，隣接するベルト領域に対応するエッジ点を削除したエッジ点集合からなる直線成分に対応しており，やはり極大性に反する．

傾きは -1 から 1 までに制限されていた．したがって，a–b 平面全体ではなく，その範囲にあるすべてのセルを調べればよいが，このように限定された領域をその領域内のセル数に線形な時間で調べることができる能力を Topological Walk はもっている．

Topological Walk では，すべてのセルを調べて直線成分として極大なものだけを報告することが可能である．これは，今までに述べた方法にはない特徴である．

実際の画像データに上記の方法を適用するときに問題となるのは縮退である．すなわち，各エッジ点をパラメータ平面のベルト領域に変換するとき，エッジ点の座標が整数値であるので，ベルト領域の境界を与える直線の傾きと y 切片はいずれも有理数となり，多数の直線が 1 点で交差することが考えられる．実際，最大では $O(N)$ 本の直線が 1 点で交差することも考えられる．幸いにも Topological Walk は縮退に強く，オーバヘッドなしに対処できることが既に筆者らによって示されている [9].

8. 密度を考慮した線分抽出

今までに述べてきた方法は，画面全体にわたって直線成分のサイズを求めることを基本にしているため，端点情報を求めたり，密度制約を取り入れることが困難である．本章では，従来と同様の投票に基づく方法でも，無理なく端点情報を求めたり，密度制約を取り入れたりできることを示す．

上に述べた投票に基づく方法ではエッジ点を順に処理したが，どのような順序でエッジ点を取り出すかについては言及していなかった．密度と端点を考慮するための基本的な考え方は，エッジ点の処理順を考慮するところにある．先にも述べたように，従来の Hough 変換法のようにすべての傾きを対象にするのではなく，水平に近い直線の検出と垂直に近い直線の抽出に分けると扱いやすい．そこで，以下では水平に近い直線，すなわち傾きが -1 から 1 までの範囲の直線の検出方法についてのみ考える．垂直に近い直線の抽出は，x 軸と y 軸を入れ替えると同じ方法で行えるからである．

さて，提案する方法では，可能な傾きの系列を定めた後，それぞれの傾き a_i，$-1 \leqq a_i \leqq 1$ について，直線成分を検出することを考える．そのために，各エッジ点 $p_j(x_j, y_j)$ について $v_j = y_j - a_i x_j$ の値を計算し，対応するバケットに投票を行う．このとき，エッジ点を x 座標の順に処理することにすれば，各バケットでは関係するエッジ点が x 座標の順に入ってくることになる．点密度とギャップを考慮するために，それぞれのバケットについて，左右の端点の座標とその間に含まれるエッジ点の数を管理しておく．これらを管理することで次のような統計データを管理できる．

count： このバケットに投票を行ったエッジ点の個数．

first： このバケットに投票を行ったエッジ点の中で現在の線分の始点と考えられる点．

局所密度（local_density）： 現在の線分の始点（first）と今回投票したエッジ点の x 座標の差でエッジ点の個数（count）を割った値．この値がしきい値より小さければ，現在投票している線分は検出しない．

ギャップ（gap）： 直前にこのバケットに投票したエッジ点から今回投票したエッジ点まで x 座標の差がどれだけあるか．このギャップが一定のしきい値以上あれば，前回のエッジ点と今回のエッジ点は異なる直線成分に属するものと判断すべきである．

これらの情報を毎回定数時間で管理する方法は自明であるので詳細は述べないが，これらの情報を用いると，各バケットにおいて投票を行ったエッジ点の集合のサイズだけではなく，連続する点の間の（x 座標の）ギャップが一定値以下であり，エッジ点の密度も一定値以上であるかどうかを管理することができる．エッジ点間の（水平方向の）ギャップが一定の（入力サイズに依存しない）しきい値を超えたり，エッジ点の密度（エッジ点の個数を x 方向の長さで割った値）が一定値以下になったときには，それまでのエッジ点の個数（count の値）によって，直線成分として出力したり，短すぎる線分として無視することができる．このとき，出力する線分の端点は，一方が first の値で管理されているエッジ点であり，他端の点は直前に入力されたエッジ点であるので，線分として出力することも容易である．より具体的には次のようになる．

［密度を考慮した線分検出アルゴリズム］

最低密度に関するしきい値 t_1 と，最大許容ギャップに関するしきい値 t_2 を定める；

傾きの系列を $(a_1, a_2, \ldots, a_M)$ とする；

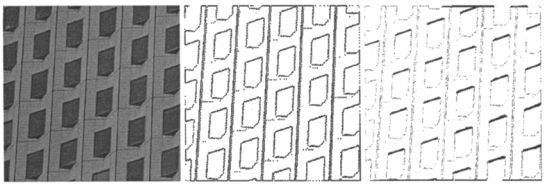

図3 実験結果
Fig. 3 Experimental results.

各バケットを初期化；

for $i = 1$ to M {

　各エッジ点 p_j について

　　$v_j = y(p_j) - a_i x(p_j)$ の値を計算する；

　　v_j をバケット B_h に入れる；

　　バケット B_h の count がしきい値以下なら次の処理をスキップし，バケットに次の値を入れる；

　　エッジ点 p_j を加えてもバケット B_h が最低密度と最大許容ギャップの制限を満たすならバケットに対応する線分を出力する；}

v_j をバケット B_h に入れるとき，以下の処理を行う．まず，バケット B_h のエッジ点数 count がしきい値以下なら，count や first の値を適当に設定して処理を終了する．バケット B_h の count が既にしきい値を超えているときは，エッジ点 p_j を加えてもバケット B_h が最低密度と最大許容ギャップの制限を満たすかどうかを判定する．もし，制限を満たすなら，まだ線分の途中であるから，これで処理を終わる．制限を満たさないときは二つの場合に分かれる．p_j を加えると制限を満たさないが，それまでは制限を満たしていた場合には，p_j の直前にこのバケットに加えられたエッジ点を線分の終点とし，first で示された点を始点とする線分を出力し，first，count 等の値を初期化する．そうでない場合には，単なる雑音成分の途中であると考えて，点 p_j を登録する以外の操作は行わない．

上記の方法を実際に C 言語で実現し，いくつかの画像に対して実験を行った．図 3 に示したのは，256 × 256 画素の jpeg 画像にエッジ検出を施した 2 値画像に対して 0 ～ 45° の線分を抽出した結果である．グレーで表したエッジ点の上から黒色で検出した線分を描いている．

9. むすび

本論文では，2 値画像に含まれる直線成分を残らず検出する問題の計算複雑度について考察した．従来か

らコンピュータビジョンの分野で用いられている投票に基づく方法は，手軽ではあるが，計算量の面では改善できない壁が存在することを示し，投票に基づかない方法の優位性を示した．更に，極大な直線成分だけを効率良く検出することができる新たなアルゴリズムも示した．本論文で述べたアルゴリズムでは，直線成分の密度や端点のコントロール，更には垂直幅ではなく直交幅を指定した場合については触れていないが，これらの問題については別の機会に報告する予定である．

謝辞　様々な御助言を頂いた日本アイビーエム東京基礎研究所の徳山豪博士に感謝する．また，本研究は文部省の科学研究費の補助を受けた．

文　献

[1] 阿部圭一，陳　風，“点の疎密性を考慮した逐次的 Hough 変換，” 情処学第 42 回全大，2D–29, 1991.
[2] T. Asano, “Digital halftoning algorithm based on a random space filling curve,” Proc. International Conf. on Image Processing, ICIP-96, pp.545–548, Lausanne, Switzerland, 1996.
[3] 浅野哲夫，計算幾何学，朝倉書店，1990.
[4] T. Asano, L. Guibas, and T. Tokuyama, “Walking in an arrangement topologically,” Int. J. of Comput. Geom. and Appl., vol.4, pp.123–151, 1994.
[5] T. Asano and N. Katoh, “Variants for the Hough transform for line detection,” Computational Geometry: Theory and Applications, vol.6, pp.231–252, 1996.
[6] T. Asano, N. Katoh, and T. Tokuyama, “A unified scheme for detecting fundamental curves in binary edge images,” Proc. of 2nd Annual European Symposium on Algorithms, pp.215–226, Springer LNCS 855, Sept. 1994.
[7] T. Asano and N. Katoh, “Variants for the Hough transform for line detection,” Computational Geomery: Theory and Applications, vol.6, pp.231–252, 1996.
[8] T. Asano, D. Ranjan, and T. Roos, “Digital halftoning algorithms based on optimization criteria and their experimental evaluation,” IEICE Trans. Fundamentals, vol.E79-A, no.4, pp.524–532, April 1996.
[9] T. Asano and T. Tokuyama, “Topological walk revisited,” IEICE Trans. Fundamentals, vol.E81-A, no.5, pp.751–756, May 1998.
[10] M. Atiquzzaman, “Multiresolution Hough transform – An efficient method of detecting patterns in images,” IEEE Trans. Pattern Anal. Mach. Intell., vol.PAMI-14, no.11, pp.1090–1095, 1992.
[11] C.M. Brown, “Inherent bias and noise in the Hough transform,” IEEE Trans. Pattern Anal. Mach. Intell., vol.PAMI-5, no.5, pp.493–505, 1983.
[12] R.O. Duda and P.E. Hart, “Use of the Hough transformation to detect lines and curves in pictures,” Comm. of the ACM, vol.15, pp.11–15, Jan. 1972.
[13] P.V.C. Hough, “Method and Means for Recognizing Complex Patterns,” U.S. Patent 3069654, Dec. 18, 1962.
[14] J. Illingworth and J. Kittler, “The adaptive Hough transform,” IEEE Trans. Pattern Anal. Mach. Intell., vol.PAMI-9, no.5, pp.690–698, 1987.
[15] 輿水大和，“Hough 直線に関する最近の研究動向（2），” 信学技報，PRU91-15, 1991.
[16] 輿水大和，沼田宗敏，“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-D-II, no.1, pp.56–65, Jan. 1985.
[17] H. Li, M.A. Lavin, and R.J. LeMaster, “Fast Hough transform,” Comput. Vision Graphics Image Processing, vol.36, pp.139–161, 1986.
[18] 松山隆司，長尾　真，“Hough 変換の幾何学的性質と直線群検出への応用，” 情処学論，vol.26, no.6, pp.1069–1078, 1985.
[19] 松山隆司，輿水大和，“Hough 変換とパターンマッチング，” 情報処理，vol.30, no.9, pp.1035–1046, 1989.
[20] 森本正志，尺長　健，末永康仁，“量子化誤差を考慮した Hough 変換，” 信学技報，PRU89-89, 1989.
[21] 森本正志，尺長　健，赤松　茂，末永康仁，“可変フィルタによる Hough 変換の高精度化，” 信学論（D-II），vol.J75-D-II, no.9, pp.1548–1556, Sept. 1992.
[22] M. Seki, T. Wada, and T. Matsuyama, “High Precision γ–ω Hough Transformation Algorithm to Detect Arbitrary Digital Lines,” Proc. SIGCV Workshop of IPSJ, CV-84-2, July 1993.
[23] I.D. Svalbe, “Natural representation for straight lines and the Hough transform on discrete arrays,” IEEE Trans. Pattern Anal. Mach. Intell., vol.PAMI-11, no.9, pp.941–950, 1989.
[24] 和田俊和，藤井高広，松山隆司，“γ–ω Hough 変換—可変標本化による ρ–θ パラメータ空間のひずみの除去と投票軌跡の直線化，” 信学論（D-II），vol.J75-D-II, no.1, pp.21–30, Jan. 1992.

（平成 11 年 3 月 26 日受付）

浅野　哲夫　（正員）

昭 52 阪大大学院基礎工学研究科情報工学専攻博士課程了．同年大阪電気通信大学工学部応用電子工学科講師．昭 63 同教授．平 9 年 4 月より北陸先端科学技術大学院大学情報科学研究科教授．計算幾何学の理論と応用に関する研究に従事．工博．著書に「計算幾何学」（朝倉書店）など．IEEE，ACM，SIAM，応用数理学会等各会員．平 5〜6 本学会アルゴリズム研究会主査．

河村　泰之　（学生員）

平 9 学習院大・理・数学中退，同年北陸先端科技大学院大学情報科学研究科博士前期課程入学，平 11 年 3 月同課程了．同年 4 月，同博士後期課程進学．計算幾何学の理論と応用に関する研究に従事．