情報処理学会論文誌 Vol. 49 No. 12 4049–4055 (Dec. 2008)

余弦三項漸化式による効率的なHough変換

沼田宗敏[†1] 輿水大和[†2]

画像のエッジ点群から直線を検出するパターン認識の有力な手法にHough変換がある．Hough変換高速化の試みの中で，基本演算部である三角関数と乗算を効率化する手法は最も実用性が高い．Koshimizuらは Fast Incremental Hough Transform 2（FIHT2）法を提案し，1回の乗算で曲線上の1点を発生させた．これは連立漸化式を用いて順次Hough曲線を近似的に計算する手法であるが，理論誤差が大きいという問題がある．その後，この漸化式を高精度化する手法が提案されたものの，ソフトウェア処理における計算コストは約2倍に膨らんだ．そこで本研究では，余弦三項漸化式を用いて厳密なHough曲線を生成する手法を提案する．この手法による1点の計算のための乗算回数はFIHT2法と同じく1回である．そして，実験により提案手法が従来の連立漸化式を用いたどの手法よりも高精度であること，計算速度もFIHT2法とほぼ同等であることを確認した．

An Effective Hough Transform Based on a Cosine-three-term-recurrence

Munetoshi Numada[†1] and Hiroyasu Koshimizu[†2]

One of the promising basis of the method for detecting straight lines from edge points is Hough transform (HT). For reducing the cost of HT, it is most effective to reduce the cost caused by the trigonometric functions and the multiplication involved in $\rho = x\cos\theta + y\sin\theta$. From this view point Koshimizu proposed the Fast Incremental Hough transform 2 (FIHT2) previously, and FIHT2 generates one point of a sinusoidal Hough curve on ρ-θ space just in 1 time of multiplication. Though FIHT2 is basically a method for approximating a Hough curve one by one calculation by using an alliance recurrence formula, unfortunately in FIHT2 the deviation between the approximated and true Hough curves becomes large. This is the reason why another algorithm was proposed for improving this precision problem, where the calculation cost swelled up at the sacrifice of more than twice of FIHT2. Then, in this paper, a new method for generating not an approximated but strict Hough curve is proposed by using a Cosine-Three-Term-Recurrence. The number of multiplications for calculating one point by the proposed method is 1 time as well as the FIHT2. Furthermore, an experiment confirms that the proposed technique is highly precise than any other methods based on the conventional alliance recurrence formula, and that the calculation cost is almost equivalent to or less than FIHT2.

1. はじめに

Hough変換は，クラスタリングされていないエッジ点群から直線を検出できる，パターン認識の有力な手法である[1),2)]．これは，パターン平面上のエッジ点をパラメータ平面上の曲線へと変換し，同一直線上にある点群の変換曲線がパラメータ平面上で交差する性質を利用する．しかし，処理時間の大きさが障害となっている．これは，次式に示すHough変換の基本式に含まれる，三角関数演算と乗算の計算コストが大きいためである．

$$\rho_n = x\cos\theta_n + y\sin\theta_n. \quad (1)$$

ここに，パラメータρはパターン平面を構成するx-y座標系における原点から直線までの符号つき距離，パラメータθは原点から直線に下ろした垂線とx軸のなす角度（垂角）である．また，パラメータρとθの添字$n = 0, 1, \ldots, K-1$は，θ方向分割数Kの番号である．

これまでHough変換を高速化するため多くの研究が行われているが，大別するとハードウェアを用いる手法，演算回数の低減を図る手法，基本演算部を効率化する手法の3種類に分けることができる．

まず，ハードウェアを用いる手法であるが，これは専用ハードウェアを用いて高速にHough変換を実行する手法である[3)]．これには，マルチプロセッサ方式や[4)]，角度分割数を2のべき乗としFast Fourier Transform（FFT）のようにバタフライ演算を実行する多段パイプライン方式[5)]，COordinate Rotation DIgital Computer（CORDIC）による反復演算を用いるFPGA方式など[6)]が報告されている．しかし，これらはいずれもハードウェアに特化した手法であるため，初期投資が大がかりで，しかもソフトウェア処理で置き換えると効率化がほとんど見込めない．

第2の演算回数の低減を図る手法では，まず，ピラミッド階層化手法がある[7)]．これは，高解像度の画像から低解像度の画像までピラミッド状に階層化を行い，エッジ点数を減ら

†1 株式会社ロゼフテクノロジー
Lossev Technology Corporation
†2 中京大学
Chukyo University

すことにより，演算回数を低減させる手法である．しかし，階層化の次数が大きくなるにつれ，線分の特徴である長さが指数関数的に小さくなるため，短い線分の抽出には適さない．また，エッジ成分の勾配を用いて投票する角度パラメータ θ の範囲を制限する手法[8),9)]や，エッジ点をランダムに投票することで投票数を削減する手法[10),11)]などがある．しかし，エッジ成分の勾配はノイズの影響を受けると大きく変化し，ランダム投票でも SN 比の小さなエッジ画像に適用すると直線の誤検出が生じる．このため，これらの手法はいずれもノイズを含むエッジ画像に適用できない．

第 3 の基本演算部の計算コストを効率化する手法では，計算コストの大きな三角関数演算と乗算の計算回数を削減する．実際には，三角関数はあらかじめ計算しテーブル化しておくことにより省くことができるので，乗算回数の低減が鍵となる．この手法による最初の試みは恩田らによるもので，三角関数の周期性を利用した効率的な計算手法を提案した[12)]．これは，角度パラメータ θ を M 個のブロックに分ける手法で，1 点あたりの乗算回数を 2 回から $3/2+1/M$ 回にまで低減させた．しかし，この手法で効率を上げるには分割数 M を十分に大きくする必要があり，こうすると逆にステップ数が M 倍に大きくなってしまうという問題があった．

これに対し，漸化式を用いて近似 Hough 曲線を発生させる Koshimizu らによる PLHT 法と FIHT2 法，Tagzout らによる改良 FIHT2 法などが提案された[13)–15)]．これらは式 (1) の Hough 変換とは異なる変換関数系を用いる広義の Hough 変換で，同一関数による逆変換で直線を検出するため，検出直線には理論誤差がともなわない．また，これらの曲線はインクリメンタルに計算できるので，1 点を生成するのに必要な乗算回数がほぼ 1 回と少ない．まず，PLHT 法では Hough 曲線を区分的直線で近似した近似 Hough 曲線を用い，次に FIHT2 法では Hough 曲線に対して 2 dot 程度の偏差を持つ近似 Hough 曲線を用いる．第 3 の改良 FIHT2 法では FIHT2 法よりもハードウェア化に適した並列演算方式の漸化式を用いるが，Hough 曲線との理論誤差は FIHT2 を用いた場合に比べて 2 倍ほど大きくなる．これら広義の Hough 変換系は拡張 Hough 変換と呼ばれ，Hough 曲線との理論誤差が大きくなるにつれ角度ごとの直線検出感度に大きな粗密が生じる[16)]．そこで，中島らは近似 Hough 曲線計算の漸化式を改良し，Hough 曲線との理論誤差を 10^{-4} dot 以下まで低減した[17)]．この手法はハードウェア化に適した並列演算方式の漸化式を用いる．しかしながら，1 点あたりの乗算回数は 2 回と逆に多くなり，ソフトウェアで実行すると FIHT2 法などに比べて 2 倍の処理時間を要してしまう．

このような連立漸化式を用いた基本演算部効率化の背景から，1 点あたりの乗算回数が 1 回以下，Hough 曲線との理論誤差が 10^{-4} dot 以下，そしてステップ数が小さい，という 3 つの条件を同時に満足する Hough 変換法が求められている．そこで本論文では，このような Hough 変換法を余弦三項漸化式を用いて構築する．

以下，2 章では漸化式を用いた従来の Hough 変換法について概説する．続く 3 章では余弦三項漸化式を用いて，厳密な Hough 曲線を効率良く発生させる手法を提案する．4 章の実験では計算コストと理論誤差の分析を行い，5 章で全体をまとめる．

2. 関 連 研 究

本章では基本演算部を効率化する，従来の連立漸化式を用いた Hough 変換法について概説する．

2.1 FIHT2 法

FIHT2 法では次の漸化式を用いて，ρ_n とそのペアとなる $\rho'_n \cong \rho_{n+K/2}$ を計算する[14)]．ここで，K を θ 方向分割数とし，$n=0,1,\ldots,K/2-2$ とする．また，$\Delta\theta=\pi/K$ として，$\theta_n=n\Delta\theta$，$\varepsilon=2\sin(\Delta\theta/2)$ とする．

$$\left.\begin{aligned}\rho_{n+1}&=\rho_n+\varepsilon\cdot\rho'_n\\ \rho'_{n+1}&=\rho'_n-\varepsilon\cdot\rho_{n+1}.\end{aligned}\right\} \tag{2}$$

式 (2) を解いて次式を得る．

$$\left.\begin{aligned}\rho_n&=\frac{\rho_0\cos(\theta_n-\Delta\theta/2)+\rho'_0\sin\theta_n}{\cos(\Delta\theta/2)}\\ \rho'_n&=\frac{-\rho_0\sin\theta_n+\rho'_0\cos(\theta_n+\Delta\theta/2)}{\cos(\Delta\theta/2)}.\end{aligned}\right\} \tag{3}$$

K が十分大きければ $\Delta\theta$ や ε が十分に小さくなるので，$\Delta\theta=2\sin^{-1}(\varepsilon/2)\cong 0$，$\cos(\Delta\theta/2)\cong 1$ が成立する．このため，初期値 $\rho_0=x$，$\rho'_0=y$ とおくことにより，

$$\left.\begin{aligned}\rho_n&\cong x\cos\theta_n+y\sin\theta_n\\ \rho'_n&\cong -x\sin\theta_n+y\cos\theta_n\\ &=x\cos(\theta_n+\pi/2)+y\sin(\theta_n+\pi/2)\end{aligned}\right\} \tag{4}$$

と式 (1) の近似式が得られる．すなわち，式 (2) の漸化式を順次計算することにより，式 (1) の近似 Hough 曲線である式 (4) を得ることができる．

通常の Hough 変換では，Hough 曲線の軌跡を描く変換式も，曲線の交差点から直線を求

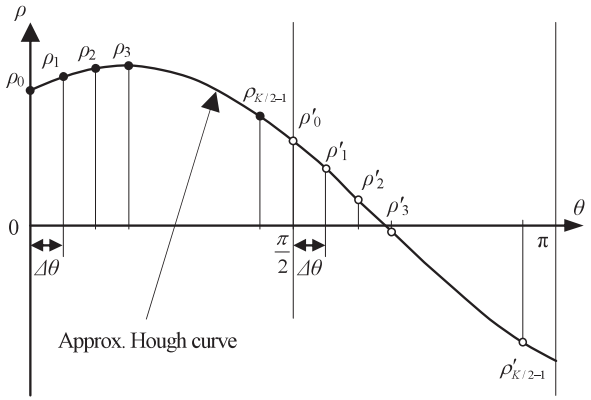

図 1　FIHT2 による近似 Hough 曲線の発生
Fig. 1　Generating the approximation Hough curve by using FIHT2.

める逆変換式も同じ式 (1) を用いる．このため，式 (1) の近似式である漸化式 (2) を用いて近似 Hough 曲線の軌跡を描き，式 (1) を用いた逆変換によって直線を求めると，変換関数と逆変換関数の違いによる理論誤差は避けられない．これに対し，同じ漸化式 (2) を用いて近似 Hough 曲線の軌跡を描いても，式 (2) と等価な式 (3) を用いた逆変換によって直線を計算すると，同一関数による逆変換であるため理論誤差は発生しない．この手法を FIHT2 法と呼ぶ．

ここで，**図 1** に FIHT2 法による近似 Hough 曲線発生の手順を示す．まず，θ 方向分割数 K を 2 の倍数におく．そして初期値を $\rho_0 = x$，$\rho'_0 = y$ とおく．続いて，式 (2) の漸化式を $K/2-1$ 回繰り返して計算することにより，近似 Hough 曲線を発生できる．ρ_n を 1 点計算するための乗算回数は 1 回である．また，Hough 曲線との理論誤差 E は ρ_n の理論誤差を E_n として，

$$\left.\begin{aligned} E_n &\cong \frac{\pi}{2K}|x\sin\theta_n| \le \frac{\pi}{2K}|x| \quad \left(0 \le \theta_n < \frac{\pi}{2}\right) \\ E_n &\cong \frac{\pi}{2K}|y\cos\theta_n| < \frac{\pi}{2K}|y| \quad \left(\frac{\pi}{2} \le \theta_n < \pi\right) \end{aligned}\right\} \tag{5}$$

であるから[17]，

$$E \le \frac{\pi}{2K}\max(|x|,|y|) \tag{6}$$

となる．

2.2　改良 FIHT2 法

Tagzout らの改良 FIHT2 法[15] は式 (2) の 2 行目右辺第 2 項の ρ_{n+1} を ρ_n に置き換えた手法である．

$$\left.\begin{aligned} \rho_{n+1} &= \rho_n + \varepsilon\cdot\rho'_n \\ \rho'_{n+1} &= \rho'_n - \varepsilon\cdot\rho_n. \end{aligned}\right\} \tag{7}$$

式 (2) は，1 行目の ρ_{n+1} の計算が終わってからでないと 2 行目の ρ'_{n+1} が計算できない直列演算方式である．これに対し，式 (7) は 1 行目と 2 行目を同時に計算できる並列演算方式であるので，並列処理可能なハードウェアで実行するとより高速化が可能である．しかし，Hough 曲線との理論誤差は FIHT2 に比べて約 2 倍と大きくなるため，実用には適していない．

2.3　中島らの手法

中島らは式 (2) を改良し，式 (1) との理論誤差がより小さな連立漸化式を提示した．これを下に示す．

$$\left.\begin{aligned} \rho_{n+1} &= \rho_n - \frac{\delta^2}{2}\rho_n + \delta\cdot\rho'_n \\ \rho'_{n+1} &= \rho'_n - \frac{\delta^2}{2}\rho'_n - \delta\cdot\rho_n. \end{aligned}\right\} \tag{8}$$

ここに変数 δ と刻み角 $\Delta\theta = \pi/K$ との間には，

$$\sin\Delta\theta = \delta/\sqrt{1+\delta^4/4} \tag{9}$$

の関係が成り立つ．ここで，δ を 2^{-m} の形にすると $\delta^2/2$ も $2^{-(2m+1)}$ の形になるので，乗算の代わりにより高速なシフト演算を用いて計算できる．ハードウェア処理では 2 回のシフト演算を同時に処理することができるので，1 回のシフト演算を用いる場合と処理時間を等しくすることができる．さらに，改良 FIHT2 法と同様，ハードウェア化した場合には並列演算方式であるためより高速化が可能で，直列演算方式の FIHT2 に比べ約 1.5 倍の高速化が報告されている[17]．

また，Hough 曲線式 (1) との理論誤差 E は次式で計算できる．すなわち，$\delta \cong \Delta\theta = \pi/K$ であるので，

$$E \le \left\{ \left(\sqrt{1+\frac{\delta^4}{4}} \right)^{\frac{K}{2}} - 1 \right\} \max(|x|,|y|) \qquad (10)$$
$$\cong \frac{K\delta^4}{16} \max(|x|,|y|) \cong \frac{\pi^4}{16K^3} \max(|x|,|y|)$$

となる[17]．これは，FIHT2 法と比較して十分に小さい．

しかし，式 (8) をソフトウェアで実行すると，係数 $\delta^2/2$ をあらかじめ計算しておいたとしても乗算回数は 1 点あたり 2 回で FIHT2 法の 2 倍になる．このため，ソフトウェアで実行した場合には約 2 倍の処理時間を要してしまう．

3. 余弦三項漸化式による Hough 変換

従来法の連立漸化式は，いずれも 1 つの式の中に 3 つの項がある三項漸化式を用いていた．本章では余弦関数の定義式から導かれる余弦三項漸化式を用い，理論誤差のない Hough 曲線を 1 点あたり 1 回の乗算で計算する手法を提示する．

3.1 余弦三項漸化式を用いた計算手法

まず次の正弦関数の定義式を考える．

$$e^{i\Delta\theta} + e^{-i\Delta\theta} = 2\cos\Delta\theta. \qquad (11)$$

ここで，両辺に $e^{i\theta_n}$ を乗じれば次式を得る．

$$e^{i\theta_{n+1}} + e^{i\theta_{n-1}} = 2\cos\Delta\theta e^{i\theta_n}. \qquad (12)$$

上式の実数部と虚数部から，以下の一対の漸化式を得る．

$$\left.\begin{aligned} \cos\theta_{n+1} + \cos\theta_{n-1} &= 2\cos\Delta\theta\cos\theta_n \\ \sin\theta_{n+1} + \sin\theta_{n-1} &= 2\cos\Delta\theta\sin\theta_n. \end{aligned}\right\} \qquad (13)$$

式の 2 つの成分は互いに独立かつ同形で，乗数係数に余弦関数を含む．これを余弦三項漸化式 CTTR（Cosine-Three-Term-Recurrence）という．上式の 1 行目に x，2 行目に y を乗じた後に 2 つの式を加え，これに式 (1) を代入すると次式を得る．

$$\rho_{n+1} + \rho_{n-1} = 2\cos\Delta\theta \cdot \rho_n. \qquad (14)$$

ここで，$\alpha = 2\cos\Delta\theta$ と置き換えることにより，次式の三項漸化式を得る．

$$\rho_{n+1} = \alpha \cdot \rho_n - \rho_{n-1}. \qquad (15)$$

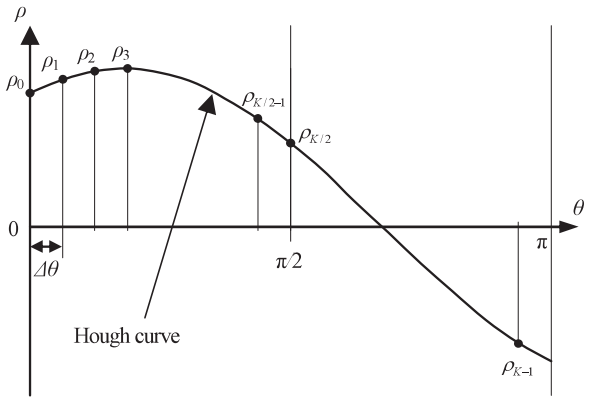

図 2　提案手法による Hough 曲線の発生（K：偶数）
Fig. 2　Generating the Hough curve by using the proposed method (K: even number).

ρ_0 と ρ_1 を初期値として与え，$n = 0$ から $K-3$ まで上式を計算すると順次 Hough 曲線が得られる．K は θ 方向分割数である．発生する Hough 曲線には理論誤差がともなわない．初期値は式 (1) から計算できるので，$\sin\Delta\theta$，$\cos\Delta\theta$ を定数化しておけば，初期値の計算も含めて三角関数の計算はいっさい不要となる．また，Hough 曲線 1 点を生成するための乗算回数も FIHT2 と同じ 1 回である．以下に，提案手法のアルゴリズムを示す．メインループのステップ数はわずか 1 行と少ない．

［Initialize］
$\Delta\theta = \pi/K;\ \alpha = 2\cos\Delta\theta$
$\rho_0 = x;\ \rho_1 = x\cos\Delta\theta + y\sin\Delta\theta$
［loop］$n = 0, 1, 2, \ldots, K-3$
$\rho_{n+2} = \alpha \cdot \rho_{n+1} - \rho_n$

ここで，本手法で発生した点列を**図 2** と**図 3** に示す．これらの点列は，FIHT2 法，中島らの手法とは異なり，厳密な Hough 曲線である．また，FIHT2 法や中島らの手法では，θ 方向分割数 K の半分ずつを一対の漸化式を用いて計算するだけでよいから，K は偶数でなければならなかった．これに対し，提案手法の漸化式は一対ではなく 1 つであるから，図 2 のように K が偶数であっても，図 3 のように奇数であってもかまわない．

3.2 計算コストなどの評価

すでに明らかにした FIHT2 法，中島らの手法および提案手法における，ρ-θ パラメータ

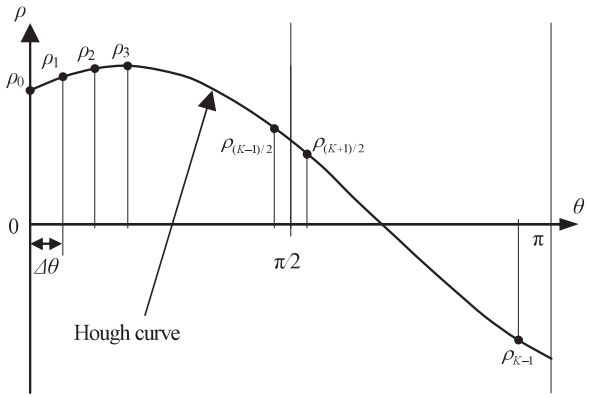

図 3　提案手法よる Hough 曲線の発生（K：奇数）
Fig. 3　Generating the Hough curve by using the proposed method (K: odd number).

表 1　手法別乗算回数と理論誤差
Table 1　Number of multiplications and the theoretical deviations.

	FIHT2	Nakashima	Proposed
Multiplications	1	2	1
Deviations (dot)	$\pi/2$	$\pi^4/(16K^2)$	0

平面上に描く軌跡の 1 点を計算するための乗算回数を，**表 1** にまとめた．FIHT2 法および提案手法では，1 回の乗算で近似 Hough 曲線または Hough 曲線の 1 点を計算できる．これに対し，中島らの手法では 2 回の計算が必要である．

また，Hough 曲線と各手法の生成曲線との理論誤差を調べるため，θ 方向の分割数 K を便宜上，エッジ点の座標最大値 $\max(|x|, |y|)$ に等しくした．また，ρ 方向の 1 dot を，x-y 座標系の画像の 1 dot と等しくした．この理論誤差を表 1 にあわせて示す．FIHT2 法の理論誤差は，$\pi/2 \cong 1.57$ dot と無視できないほど大きい．これに対し，中島らの手法の理論誤差は，式 (10) より $E = \pi^4/(16K^2)$ となり，$K > 247$ で $E < 10^{-4}$ dot となるほど十分に小さい．

4. 実　　験

本章では，FIHT2 法，中島らの手法，提案した手法の 3 手法で，Hough 曲線に対する偏差と計算速度を比較する．改良 FIHT2 法は，Hough 曲線に対する理論誤差で FIHT2 と中島らの両手法に劣るため除外した．計算には，CPU に Pentium4-1.7 GHz を搭載したパソコンを用い，プログラムは VC/C++，変数は倍精度実数で取り扱った．

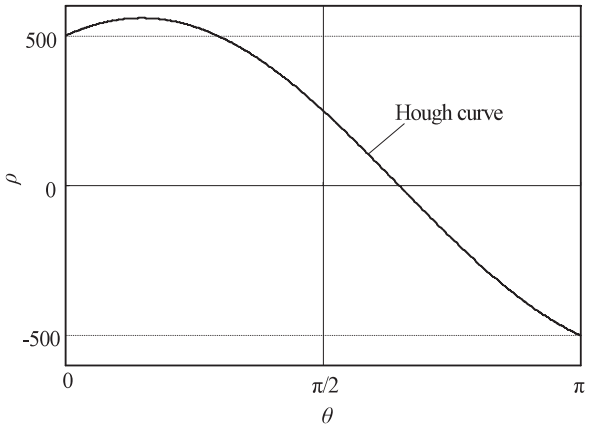

図 4　実験で用いた Hough 曲線
Fig. 4　Hough curve used in the experiment.

表 2　Hough 曲線との偏差（dot）
Table 2　Deviation to the Hough curve (dot).

	FIHT2	Nakashima	Proposed
Theory	1.57	2.44×10^{-5}	0
Experiment	1.57	2.42×10^{-5}	3.56×10^{-10}

4.1　Hough 曲線生成における偏差

画像上のエッジ点の座標を $x = 500$，$y = 250$，θ 方向分割数を $K = 500$ として，Hough 曲線または近似 Hough 曲線を発生させる．**図 4** は式 (1) で計算した，実験で用いる Hough 曲線である．これと，3 手法で計算した曲線との理論誤差を調べる．

表 2 に，Hough 曲線と各手法との偏差 E を，理論値と実測値とに分けて示す．理論値は表 1 より計算した．FIHT2 法の理論誤差は $\pi/2 = 1.57$ dot であったが，実測値も 1.57 dot となり一致した．また，中島らの手法の理論誤差は $\pi^4/(16K^2) = 2.44 \times 10^{-5}$ dot であったが，実測値も 2.42×10^{-5} dot となりほぼ一致した．なお，提案手法の理論誤差は 0 dot で

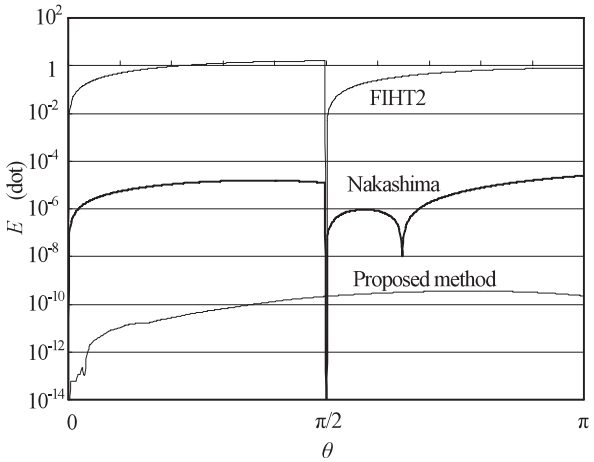

図 5　Hough 曲線と各手法との偏差 E
Fig. 5　Deviation E to Hough curve.

あるが，実測値は 3.56×10^{-10} dot となった．これは倍精度計算における丸め誤差が，漸化式の繰返し計算で累積されたもので実用上は無視できる．また，3 手法の中では最も小さい．

図 5 は，Hough 曲線と 3 手法による計算値との偏差を，$0 \leq \theta < \pi$ の範囲で示したものである．$\theta = 0$，$\pi/2$ を除くすべての区間で提案手法，中島らの手法，FIHT2 法の順で偏差が小さくなった．ただ，$\theta = 0$，$\pi/2$ では，提案手法よりも FIHT2 法や中島らの手法で偏差が小さくなった．これは，漸化式の初期値である ρ_0 と ρ_0' とが Hough 曲線上の点として直接与えられるためである．このため，偏差曲線は $\theta = \pi/2$ で不連続となる．これに対し提案手法では，ρ' を用いないので $\theta = \pi/2$ で不連続にならない．

4.2 計算時間

3 手法による計算時間を測定するため，画像上にランダムに発生させた 10 万個のエッジ点 (x, y) を用いて，これから Hough 曲線および近似 Hough 曲線を計算した．計算時間を 1 ms 単位で計測し，その平均値として Hough 曲線または近似 Hough 曲線の 1 点あたりの計算時間を求めた．なお，これにはプロット時間を含まない．

表 3 に計算時間を示す．計算時間は FIHT2 法が最も速く，次いで提案手法，そして中島らの手法の順であった．FIHT2 法と提案手法との処理時間の比率は 1.03 倍で，ほとんど同じであった．これは，どちらも 1 点あたりの乗算回数が 1 であるためである．これに対し，

表 3　曲線上の 1 点の計算時間（sec）
Table 3　Calculate cost per one point (sec).

	FIHT2	Nakashima	Proposed
Time	1.95×10^{-8}	3.16×10^{-8}	2.01×10^{-8}
Ratio	1.00	1.62	1.03

乗算回数が 2 である中島らの手法では，FIHT2 法に対して処理時間の比率は 2 でなく 1.62 となった．これは，計算時間にはループ処理に要する時間も含まれるため，乗算回数の比率だけに依存しないためである．なお，式 (1) をそのまま計算するオリジナルの Hough 曲線の 1 点あたりの計算時間は 3.10×10^{-7} sec で，提案手法に対する処理時間の比率は 15.4 であった．

5. おわりに

FIHT2 法のように漸化式を用いたインクリメンタルな計算法でありながら，近似ではなく厳密に Hough 曲線を生成できる手法を提案した．この手法の基本演算部の乗算回数は FIHT2 法と同じく 1 回であり，ステップ数もわずか 1 行と少ない．また，FIHT2 法では θ 方向分割数 K が偶数でなければならなかったが，提案手法にはこのような制限がなく奇数であってもよい．

Hough 曲線との偏差に関する実験では，提案手法は従来の 2 手法である FIHT2 法，中島らの手法のいずれよりも偏差が小さく，しかも計算誤差のみに起因する偏差は 10^{-9} dot 未満ときわめて小さかった．さらに，曲線上の 1 点の計算時間に関する実験では，提案手法の計算速度は FIHT2 法とほぼ同等であり，中島らの手法に比べて約 1.6 倍高速であった．

参考文献

1) Duda, R.O. and Hart, P.E.: Use of the Hough transformation to detect lines and curves in pictures, *Comm. ACM*, Vol.15, No.1, pp.11–15 (1972).
2) 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).
3) Hanahara, K., Maruyama, T. and Uchiyama, T.: A real-time processor for the Hough transform, *IEEE Trans. Pattern. Anal. Machine Intell.*, Vol.10, No.1, pp.121–125 (1988).
4) Ben-Tzvi, D., Naqvi, A.A. and Sandler, M.B.: Synchronous multiprocessor imple-

mentation of the Hough transform, *Computer Vision, Graphics and Image Processing*, Vol.52, No.3, pp.437–446 (1990).
5) Vuillemin, J.E.: Fast Linear Hough Transform, *Int. Conf. Application Specific Array Processors*, pp.1–9 (1994).
6) Karabernou, S.M., Kessala, L. and Terranti, F.: Real-time FPGA implementation of Hough Transform using gradient and CORDIC algorithm, *Image and Vision Computing*, Vol.23, pp.1009–1017 (2005).
7) Bongiovanni, G., Guerra, C. and Levialdi, S.: Computing the Hough transform on a pyramid architecture, *Machine Vision and Applications*, Vol.3, pp.117–123 (1990).
8) Cucchiara, R. and Filicori, F.: The vector-gradient Hough transform, *IEEE Trans. Pattern. Anal. Machine Intell.*, Vol.20, No.7, pp.746–751 (1998).
9) O'Gorman, F. and Clowes, M.B.: Finding picture edges through collinearity of feature points, *IEEE Trans. Comput.*, Vol.25, No.4, pp.449–456 (1976).
10) Xu, L. and Oja, E.: Randomized Hough transform (RHT): Basic mechanisms, algorithms, and computational complexities, *Image Understandings*, Vol.57, No.2, pp.131–154 (1993).
11) 加藤邦人，山崎　秀，遠藤利生，村上和人，鳥生　隆，輿水大和：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, No.1, pp.81–86 (1997).
12) 恩田邦夫，青木由直：三角関数の周期性を利用した Hough 変換の高速計算法，信学論（D），Vol.J70-D, No.10, pp.2009–2011 (1987).
13) Koshimizu, H. and Numada, M.: On a fast piece-wise linear Hough transform PLHT and its application, *IAPR Workshop on CV-Special Hardware and Industrial Application*, pp.335–339 (1988).
14) Koshimizu, H. and Numada, M.: FIHT2 algorithm: A fast incremental Hough transform, *IEICE Trans. Inf. & Syst.*, Vol.E74, No.10, pp.3389–3393 (1991).
15) Tagzout, S., Achour, K. and Djekouse, O.: Hough transform algorithm for FPGA implementation, *Signal Processing*, Vol.81, pp.1295–1301 (2001).
16) Koshimizu, H., Murakami, K. and Numada, M.: On a Warp Model of Hough Transform, *ACCV'93*, pp.676–679 (1993).
17) 中島勝行，矢加部英敏，大淵　豊，井上勝敬：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, pp.1509–1515 (1996).

(平成 20 年 4 月 14 日受付)
(平成 20 年 9 月 10 日採録)

沼田　宗敏（正会員）

1984 年富山大学理学部物理学科卒業．2006 年富山県立大学大学院工学研究科博士後期課程修了．博士（工学）．1984 年（株）ロゼフテクノロジーに入社し現在に至る．主たる研究テーマは，人工知能，知能機械学，コンピュータグラフィックス，画像処理．電子情報通信学会，精密工学会各会員．共著書に，『最新コンピュータグラフィックスがわかる』（技術評論社）等.

輿水　大和（正会員）

1975 年名古屋大学大学院工学研究科博士課程修了．工学博士．名古屋大学助手，名古屋市立工業研究所を経て，現在中京大学情報理工学部長・教授．画像処理，マシンビジョン，パターン認識，顔研究，画像デジタル化理論，Hough 変換等画像処理基礎理論，およびそれらの産業応用研究に従事．電気学会（上級会員），電子情報通信学会，SICE，精密工学会，日本顔学会各会員．共著書に，『画像処理基本技法』（技術評論社），『コンピュータビジョン』（近代科学社），『実践画像処理』（Springer-Verlag 東京），『信号処理』（オーム社），『環境知能のすすめ』（丸善）等．ViEW2002・ViEW2005 小田原賞（IAIP，JSPE）等受賞.