情報処理学会論文誌　数理モデル化と応用　Vol.5　No.3　107–119 (Sep. 2012)

# 最適性を保証する多重解像度表現を用いた離散直線当てはめ

宮武 孝尚[1,a)]　中力 雅人[1,†1]　清水 郁子[1,b)]

受付日 2012年4月19日，再受付日 2012年6月7日，
採録日 2012年7月9日

**概要**：コンピュータビジョンにおける重要な課題の 1 つである画像中の点に対する直線当てはめでは，一般には，画像中の点のうちどれが直線に含まれているかをあらかじめ知ることは難しいため，与えられた点群に外れ値が含まれることを考慮する必要がある．これに対し，最も多くの点が整合する最適解を求める離散直線当てはめの手法が提案されているが，計算時間がかかるという問題がある．そこで本手法では，多重解像度表現を適用し，計算の高速化を図る．このとき，最適性を保証するダウンサンプリングと直線表現を適用する．そして，解像度が深さに対応する木構造で多重解像度の問題を保持して最適解を効率良く探索する．このとき，類似度の高い点の組合せ（問題）を統合し，離散直線当てはめの動作を並列化することで計算時間の短縮を狙う．提案手法の有効性を示すために，2 次元実データのエッジを抽出したデータに対して，多重解像度表現を使用したものとオリジナルの解像度で離散直線当てはめを行い，計算時間を比較する．計算時間を比較したところ，大部分のデータにおいて計算時間が短縮されることを確認した．また，最適性を保証したことで同じ解が得られることも確認した．

**キーワード**：離散直線，直線当てはめ，多重解像度，木構造，高速化，並列処理

# Optimal Digital Line Fitting Using Multiresolutional Representation

Takahisa Miyatake[1,a)]　Masato Churiki[1,†1]　Ikuko Shimizu[1,b)]

Received: April 19, 2012, Revised: June 7, 2012,
Accepted: July 9, 2012

***Abstract:*** Line fitting to image points is one of the most important issues in computer vision. In general, image points includes many outliers, which do not belong to the line to be estimated and they cannot be eliminated beforhand. Therefore, it is necessary to detect outliers and to fit a line simultaneously. To solve this problem, discrete line fitting methods have been proposed in the literature. In these methods, the optimal solution is obtained which is the most consistent line among all possible combinations of image points. However, they require much computational costs. In this method, by applying the multiresolution representation, the computational time of the optimal discrete line fitting is much reduced. Our method guarantees the optimality by a linear expression and down-sampling which preserve the opitimality. In addition, we propose a efficient search algorithm for obtaining the optimal solution using a multi-resolution tree structure whose depth is corresponding to the resolution. And, by integrating similar problems and parallel processing of two discrete line fittings, the computation time is reduced. In order to show the effectiveness of the proposed method, experimental results of a discrete line fitting were shown by changing the initial resolution and it was confirmed that the computation time was much reduced by our method for many data sets.

***Keywords:*** digital line, line fitting, multiresolution, tree structure, acceleration, parallel operation

## 1. はじめに

直線当てはめは，コンピュータビジョンにおいて基本的かつ重要な問題の 1 つである．これは，画像上のエッジ点群，すなわち，2 次元空間上に存在する点群に最も当てはまる直線を求める問題である．直線当てはめはカメラキャ

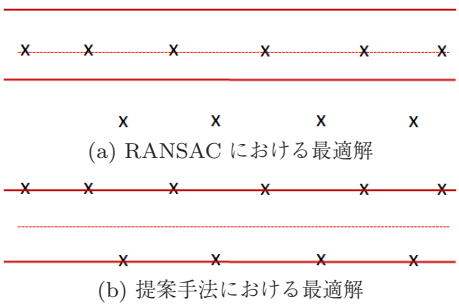
[1] 東京農工大学大学院
Graduate School of Engineering, Tokyo University of Agriculture and Technology, Koganei, Tokyo 184–8588, Japan
[†1] 現在，野村総合研究所
Presently with Nomura Research Institute
[a)] 50012646134@st.tuat.ac.jp
[b)] ikuko@cc.tuat.ac.jp

リブレーションや人工物の認識など広く用いられている．

一般に，直線当てはめにおいて，画像中のエッジ点すべてが 1 つの直線に含まれるとは限らない．そのためデータにノイズや誤差が含まれている可能性を考慮する必要があり，従来は統計的な最適化で直線当てはめを行う手法が提案されてきた．たとえば，古典的な手法の 1 つである最小二乗法 [1] では，入力点と直線との距離を最小にするようにパラメータを推定することで直線当てはめを行う．しかし，入力される点（エッジ点）が推定したい直線上にないものを含む場合（外れ値を含む場合）にはノイズが正規分布に従うという前提を満たさないため，望ましい結果は得られない．それに対し，外れ値を含むデータに対しても頑健な直線当てはめができる手法として M 推定 [2] が提案されている．M 推定における誤差関数では，一定以上の誤差を持つ点の重みを低くすることで外れ値の影響を受けにくくしている．しかし，この手法であっても外れ値の割合が多い点群に対しては正しい結果が求められないという問題がある．また，統計的な手法である RANdom SAmple Consensus（RANSAC）[3] は非常によく用いられる手法である．この手法はランダムに 2 点をサンプリングし，その 2 点を通る直線を求め，残りの点の直線からの距離を計算し，直線からの距離が閾値以内の点の数を数えることを繰り返す．十分な回数のサンプリングを繰り返し，最も多くの点が直線から閾値以内になったときの直線を解とする．しかし，RANSAC では最適解が得られることは保証されていない．これは，RANSAC が確率的なサンプリングによっているためだけではない．RANSAC により真の直線が求まる可能性があるのは，入力データに 2 点以上の誤差を含まないデータが含まれる場合のみである（**図 1**）．

一方で，離散幾何学においても直線当てはめが研究されている．離散幾何学では点は整数値で表現され，直線は一定の幅を持つを 2 本のユークリッド直線の間にある整数値の点の集合として定義される．これを離散直線と呼ぶ．従来，与えられた点群が離散直線かどうかを判定する手法 [4], [5], [6], [7]，与えられた点群が離散直線である場合に直線の幅を計算する手法 [8], [9] が提案されている．しかしこれらの手法では，点集合に外れ値を含むことを想定していない．これに対し，外れ値が含まれる場合に，あらゆる可能な点の組合せの中から直線に含まれる点の数が最大になるような組合せを求める手法 [10], [11] が提案されている．これらの手法では，一定の幅の直線に対して含まれる点の数が最大になるという意味での最適解を求めることができる．文献 [11] では，離散直線の最適解を求める問題を混合整数計画問題として定式化しているが，非常に計算時間がかかっていた．文献 [10] の手法では，双対変換を行いパラメータ空間においてトポロジカルスイープを適用することにより入力点の数 $N$ に対し $O(N^2 \log N)$ の計算時間で最適解を求めることができる．しかしながら，点の数が増えると計算時間が増大するため，多重解像度表現により明らかな外れ値を低い解像度で除外することにより計算時間を削減する手法 [12] が提案されたが，この手法では離散直線のモデルや多重解像度表現の問題で解の最適性が保証されていなかった．

そこで本論文では，多重解像度表現により離散直線の最適解を高速に求める手法を提案する．本論文では，4 連結と呼ばれる離散直線のモデルでは，離散直線をダウンサンプリングしたものは必ず離散直線であるという性質 [13] を用い，低い解像度で求められた解すべてを高い解像度で解くことで最適性を保証する．このとき，多重解像度で得られた解を木構造で保持して枝刈りを行い，類似した解を統合することで効率良く探索する．さらに，独立な離散直線探索を並列化して実行することで計算時間を短縮する．



(a) RANSAC における最適解

(b) 提案手法における最適解

**図 1**　2 つの直線当てはめ手法における最適解

**Fig. 1**　Optimal solution of line fitting by two methods.

## 2.　離散直線と離散直線当てはめの定義

以下では $\mathbb{R}$ を実数の集合，$\mathbb{Z}$ を整数の集合とする．

### 2.1　離散直線の定義

まず，2 次元ユークリッド空間 $\mathbb{R}^2$ における直線 $L$ を以下のように定義する．

$$L = \{(x, y) \in \mathbb{R}^2, : ax + by + c = 0\} \tag{1}$$

$a, b, c \in \mathbb{R}$ である．次に 2 次元離散空間 $\mathbb{Z}^2$ における離散直線 $D(L)$ を以下のように定義する．

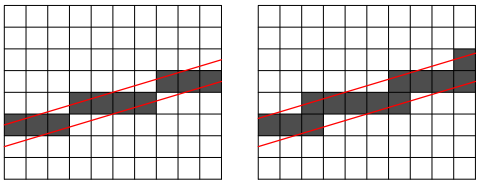
$$D(L) = \{(x, y) \in \mathbb{Z}^2 : 0 \leq ax + by + c < w\} \tag{2}$$

$a, b, c, w \in \mathbb{R}$ である．

離散直線に含まれる座標値 $(x, y)$ は整数値しかとらないが離散直線のパラメータは実数とする．また，$w$ は幾何学厚み（geometrical distance）と呼ばれ，文献 [8] によると幾何学的厚みに従って離散直線は次の 4 種類に分けられる．

- **naive:** $w = \max(|a|, |b|)$，8 連結（**図 2** (a)）
- ***-connected:** $\max(|a|, |b|) < w < |a| + |b|$
- **standard:** $w = |a| + |b|$，4 連結（図 2 (b)）
- **thick:** $w > |a| + |b|$，4 連結

図 2 に幾何学的厚み $w$ による離散直線の違いを示す．本



(a) 8 連結（naive）な離散直線　(b) 4 連結（standard）な離散直線

図 2　幾何学的厚み $w$ による離散直線の違い

**Fig. 2**　The difference between the discrete linear geometric thickness by $w$.

手法では $w = |a| + |b|$ となる離散幾何学では standard と呼ばれる 4 連結のモデルを使用する．本論文では standard な離散直線のモデルのことを 4 連結と呼ぶ．離散幾何学においては 4 連結という表現は thick と呼ばれるモデルも含むが，本論文中では standard なモデルのみを示すこととする．

### 2.2　離散直線当てはめの定義

前節での離散直線の定義から離散直線当てはめを次のように定義する．与えられた離散点の集合を $S$ とする．すなわち，

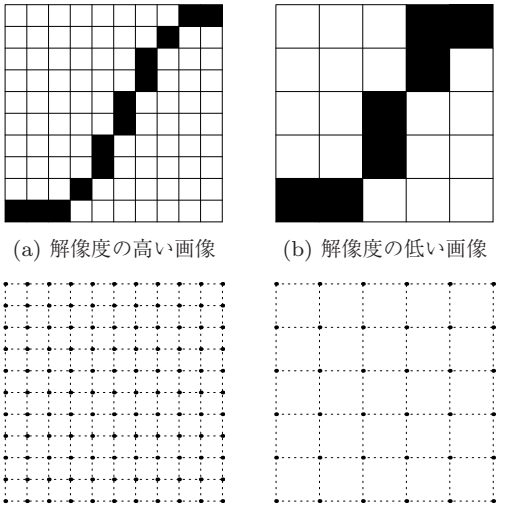
$$S = \{\boldsymbol{x}_i \in \mathbb{Z}^2 : i = 1, 2, \ldots, N\} \tag{3}$$

を離散点の集合 $S$ と定義し，$S$ に含まれる点を最も多く含む離散直線 $D(L)$ を求めるということである．ここで $\boldsymbol{x} \in S \cap D(L)$ となるような点 $\boldsymbol{x} \in S$ が内部点であり，それ以外の点が外れ値である．離散直線当てはめの従来手法 [10] では，双対変換を行いパラメータ空間においてトポロジカルスイープを適用することにより入力点の数 $N$ に対し $O(N^2 \log N)$ の計算時間で最適解を求めることができる．

## 3.　多重解像度表現とダウンサンプリング

本手法では，あらかじめ低解像度で明らかに外れ値である点を除外し，効率良く探索を行う．低解像度での計算で得られた結果を高解像度での計算に反映させることを繰り返すことで計算時間を短縮するために，複数の解像度で表現することを，多重解像度表現と呼ぶ．

多重解像度表現は従来より画像処理の分野で性質が研究され幅広く利用されてきた（たとえば，文献 [14], [15]）．これらの手法では，画像を連続信号ととらえ，解像度の低い画像はガウス関数でぼかすことにより得て，低い解像度で得られた解を高い解像度に反映させることが行われてきた．一方，近年では文献 [13] のように離散幾何学における多重解像度表現も研究されるようになってきた．本手法では，離散幾何学のための多重解像度表現を利用することで，最適性を損なうことなく直線当てはめの計算時間を短縮することを狙う．このとき，最適性の保証のため，異なる解像度で得られる解の同一性を保証するモデルを導入する．



(a) 解像度の高い画像　(b) 解像度の低い画像　(c) $\mathbb{Z}^2$ の点集合の一部　(d) $S(2,2)$ の点集合の一部

図 3　ダウンサンプリング

**Fig. 3**　Downsampling.

最適性を保証するためには，低解像度での離散直線が高解像度での離散直線を必ず含んでいる必要がある．そこで，本手法では文献 [13] でこれを保証することが示されているオリジナルの解像度の画像から低解像度の画像を得るためのダウンサンプリングと 4 連結の離散直線モデルを用いる．

まず，低い解像度のグリッドを意味する $\mathbb{Z}^2$ の部分集合 $S(h,v) = \{(Xh, Yv) : \forall (X,Y) \in \mathbb{Z}^2, h, v \in \mathbb{Z}\}$ を考える（図 3 (d)）．ここで $h$，$v$ は元の解像度に対する低い解像度グリッドにおける $x$，$y$ 方向の解像度の粗さを示している．たとえば解像度 1/2 の低解像度グリッドを考えるのであれば $h = v = 2$ となり解像度が 1/4 のものを考えるのであれば $h = v = 4$ となる．通常は低解像度のグリッドは正方形であると考えるので $h = v$ となる．$[0,h) \times [0,v)$ は基本領域であり，ベクトル $X(h,0) + Y(0,v)$ によって平行移動させることでタイル状に並べられ，それぞれのタイルは $S(h,v)$ の点を必ず 1 つ含む．ここで $D(L)$ と $S(h,v)$ の共通部分に着目する．これを，部分集合 $S(h,v)$ における $D(L)$ のダウンサンプリングとする．図 3 (a) と図 3 (d) の共通部分が図 3 (b) といえる．

ここで，$\left[\frac{x}{h}\right]$ を $x$ を $h$ で割った商，$\left\{\frac{x}{h}\right\}$ をその除算の余りとすると，$\mathbb{Z}^2$ 上で $S(h,v)$ によって生成された新しい座標系 $(X,Y)$ は次のようになる．

$$X = \left[\frac{x}{h}\right] \qquad Y = \left[\frac{y}{v}\right]$$
$$x = hX + \left\{\frac{x}{h}\right\} \qquad y = vY + \left\{\frac{y}{v}\right\}$$

文献 [13] によると，4 連結な離散直線をダウンサンプリングしたものは 4 連結な離散直線になることが証明されている．すなわち，オリジナルの点集合をダウンサンプリングした点集合には，必ずオリジナルの点集合における最適

解を含むような 4 連結な離散直線が存在するといえる．つまり，低解像度で得られる 4 連結な離散直線は，オリジナルの解像度での 4 連結な離散直線を含む．ここで，低解像度での離散直線のうち，最も内部点が多かったものが必ずしもオリジナルな解像度の最適解であるとは限らないことに注意しておく．しかし，低解像度で得られた離散直線のうち，どれかは必ずオリジナルな解像度での離散直線の最適解を含んでいる．

## 4. 提案手法

### 4.1 離散直線探索の手法

与えられた点群から離散直線を見つける離散直線当てはめを行う手法として，トポロジカルスイープに基づく手法 [10] を適用する．この離散直線探索の手法は，最適解だけではなく，すべての組合せの解を得ることができる．そのため，最適性を保証するダウンサンプリングを行い，低解像度ですべての離散直線の解を得ることができ，低解像度で得られた解で内部点の候補を絞り，高解像度で再度探索を行うことで，最適性を保証しつつ効率良く離散直線当てはめが可能となる．なお，文献 [10] では 4 連結な離散直線ではなく naive と呼ばれる離散直線モデルを使用しているが，最適性保証の観点から本手法では 4 連結な離散直線に変更する．

4 連結な離散直線モデルでは $w = |a| + |b|$ であるので，離散直線の式は次のようになる．

$$\mathbf{D}(\mathbf{L}) = \{(x, y) \in \mathbb{Z}^2 : 0 \leq ax + by + c < |a| + |b|\} \quad (4)$$

ここで，$|a| + |b| = 1$ に正規化すると，$|b| = 1 - |a|$ である．$b \geq 0$ のとき，$b = 1 - a$（$a \geq 0$）または $b = 1 + a$（$a < 0$）となり，法線ベクトルは 1 つの直線につき 2 つあるので，$b < 0$ の場合は計算しなくてよい．それぞれの場合を式 (4) に代入すると，次のようになる．

$$\mathbf{D}(\mathbf{L}) = \{(x, y) \in \mathbb{Z}^2 : 0 \leq ax + (1 - a)y + c < 1\} \quad (a \geq 0)$$

$$\mathbf{D}(\mathbf{L}) = \{(x, y) \in \mathbb{Z}^2 : 0 \leq ax + (1 + a)y + c < 1\} \quad (a < 0)$$

これを整理して，双対空間で考えると，

$(a \geq 0)$ :

$$\mathbf{D}(\mathbf{L}) = \{(a, b) \in \mathbb{R}^2 : 0 \leq (x - y)a + y + b < 1, 0 \leq a \leq 1\} \quad (5)$$

$(a < 0)$ :

$$\mathbf{D}(\mathbf{L}) = \{(a, b) \in \mathbb{R}^2 : 0 \leq (x + y)a + y + b < 1, -1 \leq a < 0\} \quad (6)$$

が得られる．

これら 2 つのモデルは表現したい直線の傾きにより使い分けられる．しかし，与えられた点群の直線の傾きはあらかじめ知ることはできないため，1 つの点群に対してそれぞれのモデルで離散直線当てはめを行う必要がある．よって，1 つの点群に対し，2 つのモデルを用いて離散直線探索を行うことになる．

### 4.2 多重解像度表現の適用と処理の流れ

多重解像度表現を用いる場合には，まずダウンサンプリングの定義に従い，入力点をある解像度まで下げる．このある解像度を初期解像度と呼ぶことにする．初期解像度は現段階ではヒューリスティックに指定している．そして，解像度を下げた点集合について，離散直線当てはめを行う．このときに，最適解だけではなく得られるすべての解を保存しておくことで，一部の解を破棄することなく，最適性を保証することができる．

ここで，本手法では多重解像度探索における最適性が保証されているため，どの解像度から探索を開始しても，得られる解は同一である，すなわち，最適解が得られることに注意しておく．ただし，計算時間は探索を開始する解像度により異なる．

多重解像度表現による解の絞り込みを説明する．たとえば，**図 4** の最も低い解像度では，図中の黒い点が内部点，ピンクの点が外れ値という直線が得られたとする．すると，2 番目の解像度ではピンクの点に相当する点群はあらかじめ外れ値であることが分かっているので，入力から除外できる．そして，2 番目の解像度では図中の黒い点が内部点，水色の点が外れ値であるという直線が得られると，最も高い解像度での入力は，ピンクの点および水色の点を除外した点群になる．こうすることで，低い解像度で，粗く離散直線当てはめが行われ，明らかな外れ値が除かれるため，高い解像度での入力点が少なくなる．

これを，解像度を上げながら得られたすべての解について繰り返し行うと，最終的にはオリジナルの解像度で離散直線当てはめを行うことになるが，このときに入力とされ

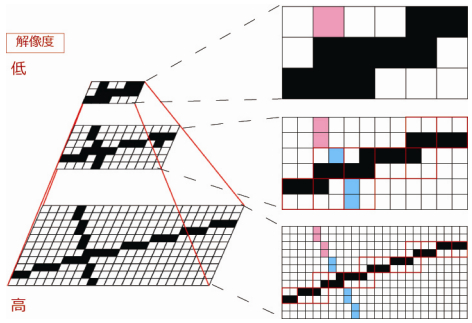



図 4　多重解像度表現による離散直線の探索

**Fig. 4**　Descrete line recognition using multiresolutional representation.

る点は，低い解像度での離散直線当てはめの結果をもとに限定された点集合であるため，オリジナルの解像度で解くよりも入力点数は少なくなり，計算時間が削減されるはずである．

具体的には，点の数を $N$ としたときに，2 次元におけるトポロジカルスイープを用いた離散直線当てはめの計算量は $O(N^2 \log N)$ である．このとき，点の数 $N = 1{,}248$ としてオリジナルの解像度で解くことを考える．使用するトポロジカルスイープでは 2 つの条件分け（離散直線のモデル）があるので，$1{,}248^2 \times \log 1{,}248 \times 2 = 3.2 \times 10^7$ となる．次に初期解像度 1/4 で多重解像度表現を用いて解く場合を考える．ここでオリジナルの解像度における最適解を 732 とする．まず，解像度 1/4 で点の数がオリジナルの解像度での点の数を 1/4 にした $N_{1/4} = 312$ になり，最適解 250（オリジナルの解像度での最適解 732 を含む）が得られたとする．次に解像度 1/4 での結果を反映させたときの解像度 1/2 の点の数は 250 の 2 倍である $N_{1/2} = 500$ になったとする．ここで同様に最適解 316 が得られ，解像度 1/1 の点の数が 316 の 2 倍である $N_{1/1} = 732$ になるとする．ここで枝刈りによって計算が終わるとすると，

$$312^2 \times \log 312 \times 2 + 500^2 \times \log 500 \times 2 + 732^2 \times \log 732 \times 2 = 1.6 \times 10^7$$

となる．この枝刈りの方法について詳しくは後述するが内部点の数が少ない問題は最適解になりえないため解かずに破棄するものである．これ以外にソートや重複の削除などの処理に必要な計算時間が加算されるが，理論的には早くなる可能性が十分期待される．

ここまでの計算量は外れ値が低解像度でうまく取り除かれ，さらに枝刈りが起こる場合を考えているが，最悪の場合での計算量についても考える必要がある．ここでの最悪の場合とは，外れ値が低解像度で取り除かれず，枝刈りが起こらないという場合である．しかし，この 2 つは同時には起こりえない．なぜなら，外れ値が取り除かれなかった場合，オリジナルの解像度ですべての入力点に対し離散直線当てはめが行われることとなり，そこで確実に最適解が求まるからである．そのため，この 2 つの事象をそれぞれ別に考える．まず，外れ値が低解像度で取り除かれない場合を考える．このとき同様に点の数 $N = 1{,}248$，初期解像度 1/4 で多重解像度表現を用いて解くと考える．解像度 1/4 で点の数がオリジナルの解像度での点の数を 1/4 にした $N_{1/4} = 312$ になり，最適解 312 が得られたとする．すなわち，すべての点が内部点だと判断されたということである．次に解像度 1/4 での結果を反映させたときの解像度 1/2 の点の数は 312 の 2 倍である $N_{1/2} = 624$ になったとする．ここで同様に最適解 624 が得られ，解像度 1/1 の点の数が 624 の 2 倍である $N_{1/1} = 1{,}248$ になるとする．ここで確実に最適解が求まるので

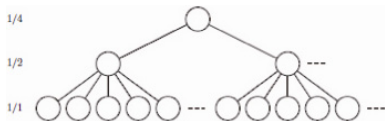



**図 5**　多重解像度表現を適用した離散直線当てはめの木構造
**Fig. 5**　Tree structure of a discrete straight line fitting to apply the multiresolutional representation.

$$312^2 \times \log 312 \times 2 + 624^2 \times \log 624 \times 2 + 1{,}248^2 \times \log 1{,}248 \times 2 = 4.0 \times 10^7$$

となりオリジナルの解像度で解くよりも大幅に計算量が増加してしまう．次に枝刈りが起こらない場合を考える．枝刈りが起こらない場合はすべての問題について解く必要があるため解像度 1/2 での動作と解像度 1/1 での動作を問題の数だけ繰り返す必要がある．ただし，入力点の数は同数であるか次第に減少する．これは内部点の多い順に問題を解くためである（問題を解く順序についての詳細は後述する）．ここでは簡単のため問題の持つ内部点がすべて同数の場合を考える．まず，最初に考えた場合と同様の挙動を行い，オリジナルの解像度で枝刈りが起こらないとする．解像度 1/2 での問題の総数が $M_2$，オリジナルの解像度での問題の総数が $M_1$ であるとき計算量は

$$\begin{aligned} &312^2 \times \log 312 \times 2 + (500^2 \times \log 500 \times 2) \times M_2 \\ &+ (732^2 \times \log 732 \times 2) \times M_1 \\ &= 1.6 \times 10^6 + 4.4 \times 10^6 \times M_2 + 1.0 \times 10^7 \times M_1 \end{aligned}$$

となる．この計算量は問題の数が重要になるが，問題数はデータにより様々である．ここで入力点は 1,248 としているため $M_2 = 2{,}000$，$M_1 = 4{,}000$ として考えると

$$\begin{aligned} &= 1.6 \times 10^6 + 4.4 \times 10^6 \times 2 \times 10^3 + 1.0 \times 10^7 \times 4 \times 10^3 \\ &= 4.9 \times 10^{10} \end{aligned}$$

となりこちらの場合もオリジナル解像度で解いた場合よりも計算量が大幅に増加してしまう．このように最悪の場合は多重解像度表現を使用することで計算量が増加するが，外れ値が低解像度で取り除かれない場合，いいかえると低い解像度ですべての入力点が 1 つの離散直線を構成することとなるようなデータは現実的なものでなく，入力点は取り除かれ方の大小はあってもほぼすべての場合で多重解像度表現を使用することで減少すると考えられるため，この場合における計算量が現実に出るとは考えにくい．また，枝刈りが起こらない場合であるが，これは枝刈りを説明する節で詳しく述べる．

このとき，探索を行う順番が重要である．直線当てはめを行うべき点群は，**図 5** のように，初期解像度での問題を根とした木構造に保持し，効率良く探索を行う．

### 4.3　問題統合

離散直線当てはめの最適性を保証するためにはすべての

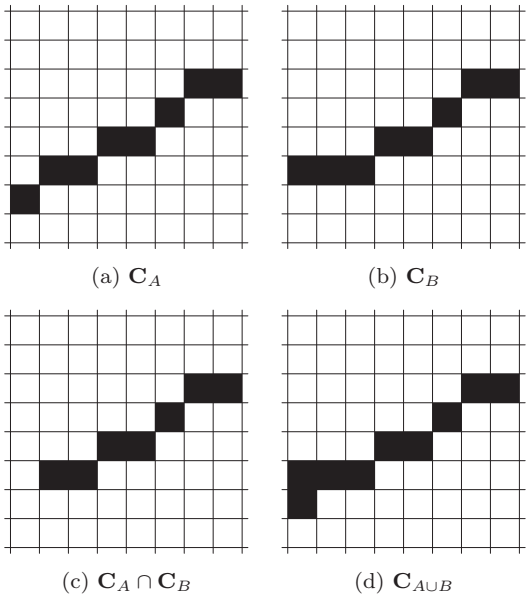



図 6　似たような解の例とその統合
**Fig. 6**　Example of similar lines and its integration.

点の組合せについて考える必要があるが，問題をすべて保持し，すべての問題に対して離散直線探索を行うと計算時間と使用メモリが莫大なものになる．さらに，低い解像度で得られる解は，互いに非常に類似した点集合であることが多く，重複して探索を行うことで計算時間が増大する．そこで計算時間とメモリ使用量を削減するためによく似た問題を 1 つの問題として扱う問題統合を行う．

ある解像度で得られた解を内部点の数でソートすると，その解像度での最適解と 2 番目の解を比べるとほとんど差異がない場合がある（**図 6**）．つまり，内部点の組合せがほとんど同じだったり，最適解の点を 1 つだけ除いたような解が 2 番目に得られたりする．似たような点群をそのまま次の解像度の入力とすると，次の解像度では同じような入力点の問題を何度も解くことになり，非常に効率が悪い．そこで，解 $\boldsymbol{C}_A$ と解 $\boldsymbol{C}_B$ に対し，共通部分 $\boldsymbol{C}_A \cap \boldsymbol{C}_B$ が，それぞれの解の点の数の閾値以上あるとき，これらの問題を統合する．これにより，共通の点を内部点として持つ似たような問題は 1 つとして扱われ，何度も重複して解くことがなくなる．また，得られた解 $\boldsymbol{C}_A$ と $\boldsymbol{C}_B$ について，それらを統合する解 $\boldsymbol{C}_{A\cup B} = \boldsymbol{C}_A \cup \boldsymbol{C}_B$ は，当然 $\boldsymbol{C}_A \subseteq \boldsymbol{C}_{A\cup B}$ かつ $\boldsymbol{C}_B \subseteq \boldsymbol{C}_{A\cup B}$ であるから，得られるべき解を失うことはない．これらのことから，最適性を損ねることなく，解くべき問題数の増加を抑えることにより計算時間とメモリ使用量を削減できることが分かる．

本論文ではそれぞれの問題の共通部分を問題の類似率と呼ぶ．ここで問題統合は類似率の閾値の設定方法が重要になる．閾値が低くなりすぎたり，高くなりすぎたりすると，問題統合による計算時間短縮が効率的に行われない．そこで我々は適応的に閾値が変化するような閾値を設定する．このとき，得られた解を内部点の数でソートすると，似たような点集合は，連続して出現することに着目している．まず，統合の閾値の初期値はヒューリステックに設定する．そして，その閾値で問題を統合していき，直前に統合された問題どうしの類似率を保持する．統合が起こらなかった場合には，そこから点集合が大きく変化していると仮定し，閾値を変更する．すなわち，それまでに統合された問題の類似率のうち最低の類似率を次の統合閾値とする．

統合閾値が低くなりすぎると入力点数が増大するので，計算時間が増加する危険性がある．一方で，統合閾値が高すぎると解くべき問題数が増加し，計算時間が増加する．前者に比べ，後者の弊害は大きい．したがって，本手法では閾値が比較的低くくなるよう閾値 −5%の範囲内の類似率の問題は統合するようにする．この提案手法で多くの問題の場合に適切な問題統合が行えるようになり，問題数の増加が抑えられ，計算時間が抑えられることが期待できる．

### 4.4　幅優先探索と枝刈り

図 5 のように木構造になった多重解像度表現を用いた離散直線当てはめ問題は，各階層における問題は，入力の点の数の降順にソートしておき，幅優先探索で探索する．このとき，明らかに最適解でない問題のノードは枝刈りを行う．

まず，最も低い解像度でのすべての点を入力とし，離散直線当てはめを行う．そして，解像度を上げながら問題を解いていき，オリジナルの解像度の問題を 1 つ解いたときに，その最適解のオリジナルの解像度相当の内部点の数が得られる．このオリジナルの解像度相当の内部点の数に対して，これから解く離散直線当てはめの問題の入力点のオリジナルの解像度相当の点の数が少ない場合は，この問題を枝刈りでき，解く必要がない．なぜなら，解像度を上げながら解いた場合にも，オリジナルの解像度相当の点の数で換算すると，その数が増えることは絶対にないからである．つまり，オリジナルの解像度で見れば，解像度を上げるということは離散直線の幅を小さくすることであり，もともと離散直線に入っていなかった点が増えることはない．幅を狭くしても外れ値になる点がない場合には，点の数が同じということはありうるが，増えるということはない．つまり，まだ解いてない離散直線当てはめの問題を解いたところで，オリジナルの解像度で解いた内部点の数より少ない入力点であれば，結果は必ずその解より少ないといえる．

**図 7** の例で説明する．左下の点の数 55 の問題を解いて最適解 52 が得られたとすると，点の数が 52 より少ない太い線の問題はすべて枝刈りできる．この枝刈りにより最適解が失われる可能性がないことは図 7 を見ても確認できる．

本手法の計算時間は枝刈りによる問題の削減が機能するか否かで大きく変化する．枝刈りはオリジナルの解像度で

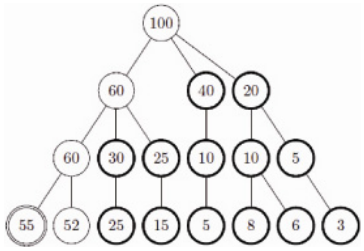



図 7　枝刈り（数字はオリジナルの解像度相当の点の数）
**Fig. 7**　Pruning (the number is the number of points equivalent to the resolution of the original).

解が求まった段階で，その解より内部点の数が少ない問題が取り除かれることで行われる．このとき枝刈りされる問題の数はその解が持つ内部点の数に依存する．そのため，内部点が少ないものが最適解になるような点群，すなわち小さな離散直線で構成されるような点群には枝刈りを行った場合でも多くの問題が残ることとなり枝刈りがうまく機能しにくいといえる．また，本手法では内部点の多い順に問題を解くため，最適解やそれに近い解を含む問題が低い解像度で内部点が非常に少なくなる場合，枝刈りに使用される解は内部点の数が非常に少なくなることが多い．これは，ある解像度までは内部点が多いと判断された問題がオリジナルの解像度に近づくにつれ外れ値の集合，もしくは内部点の少ない離散直線の集合だと認識されるためである．そのため枝刈りで問題数がほとんど削減できないことも起こる．加えて，オリジナルの解像度で最適解を含む問題が解かれるタイミングも遅くなり，枝刈りによる計算の終了が遅くなり，計算時間が効果的に削減できるとはいえない．こちらも，最適解に含まれる内部点の数が点群の数に比べ非常に少なくなる場合に起こると考えられる．逆に最適解に含まれる内部点の数が点群の数に対して比較的多い場合は，最適解が含まれる問題が優先的に解かれるようになり，また枝刈りでも多くの問題が削減でき，計算時間を大幅に短縮できると考えられる．以上のことから枝刈りは点群の数に対する最適解の内部点の数や含まれる離散直線の数に大きく依存するといえる．

### 4.5　離散直線探索の並列化

本手法では入力となる点群に対し，式 (5) と式 (6) の 2 つの離散直線のモデルを用いて離散直線を探索する．この 2 つのモデルは独立であり，片方のモデルでの結果がもう一方に影響を及ぼすことはない．そこで我々は 2 つのモデルによる離散直線探索は並列に実行する．まず，親プロセスでそれぞれのモデルでの離散直線認識を行う子プロセスを 2 つ作成する．子プロセスはそれぞれのモデルで多重解像度表現を使用し，離散直線認識を行い解を求める．親プロセスはその間，待ち状態となって子プロセスが終了するのを待つ．子プロセスは解が求まり次第，親プロセスに解を送りプロセスごと消去される．親プロセスは 2 つの子プロセスから解を受け取るとより良いものを選び最適解として出力する．なお，この並列化はマルチコア CPU のみで行っている．

### 4.6　アルゴリズム

離散直線当てはめ全体のアルゴリズムを述べる．あらかじめ，オリジナルの解像度での解を入れる解リストと，これから解くべき問題を入れる問題リストを作っておく．問題リストの問題が持っておくべき情報は，解像度，直線のモデルの種類，内部点の集合の 3 つである．まず親プロセスのアルゴリズムを示す．

- 初期問題を 2 つの直線のモデルそれぞれについて問題を生成する．この初期問題は，解像度は初期解像度に設定し，すべての点が内部点である．
- 作成された 2 つの問題についてそれぞれ子プロセスを作成する．
- 親プロセスは 2 つのプロセスが終了するまで待機する．
- 2 つのプロセスが終了し解が送られてきたらより良いものを最適解として出力する．

次に子プロセスのアルゴリズムを示す．

- 問題リストが空でない間，問題リストから最も入力点の数が多い問題を 1 つとってくる．
- 問題リストをもとに，入力となる点を限定し，解像度を落とした点集合を生成する．
- 生成した点集合に対して，問題を解く．
- 解いた問題の解像度が 1/1 であり，解がすでにあるものより良いものであれば，解を解リストに追加する．それ以外の場合は，問題リストに追加する．このとき，重複は削除し，似た問題は統合し（問題統合），点の数の降順にソートしておく．
- 解リストに解を追加したときは，その内部点の数より入力点の数が少ない問題を，問題リストから削除する（枝刈り）．
- 問題リストが空になるまで繰り返し，空になったら親プロセスに解を送りプロセスを破棄する．

## 5.　実験と考察

提案手法を評価するため，実画像からエッジを抽出したデータに対して離散直線当てはめを行った．使用したデータの一例を**図 8** に示す．これは OpenCV [16] のサンプルに含まれるチェッカパターンに対しエッジ抽出を行い，細線化を行ったデータである．

初期解像度を変更して離散直線当てはめを行い計算時間を比較した．結果を**図 9** に示す．

加えて，離散直線探索の並列処理についての効果を調査するために，並列処理を行ったものと並列処理を行っていないもので計算時間を比較する．その結果を**図 10** に示す．

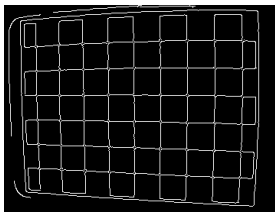


図 8　実験に使用したデータの一例
**Fig. 8**　An example of the edge points for experiments.

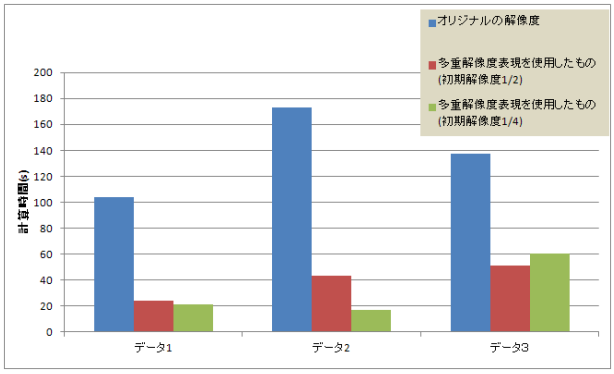



図 9　離散直線当てはめの計算時間の比較
**Fig. 9**　Comparison of computation time of digital line fitting.

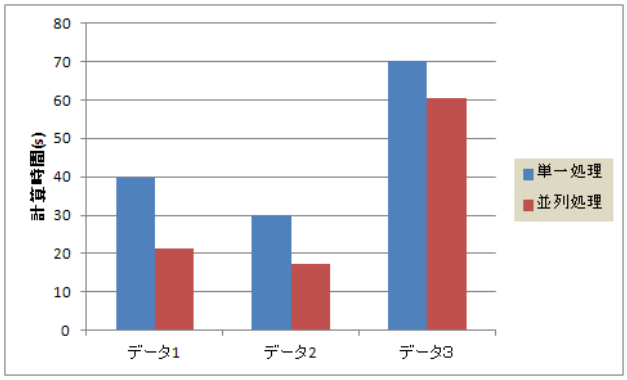
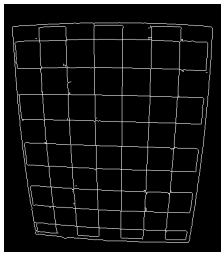


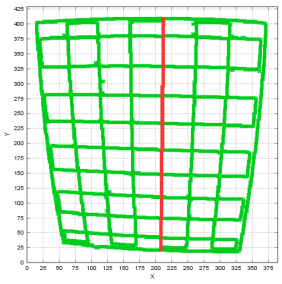

図 10　離散直線当てはめの並列処理による計算時間の比較
**Fig. 10**　Comparison of computation time by parallel processing of digital line fitting.

すべてのデータにおいて，多重解像度表現を使用したものは，オリジナルの解像度で解いたものと同様の最適解が得られていることを確認した．いずれの結果においても，提案手法である多重解像度表現を適用した離散直線当てはめによって，計算時間が短縮されていることが示された．

しかし，データ 3 は初期解像度が粗くなったときに計算時間が増大する結果となった．これは，低い解像度において，オリジナルの解像度での最適解の離散直線の点群を含む離散直線が，低い解像度での最適解となる，もしくは最適解に近いものであれば本手法は最大の効果を発揮するためであると考えられる．データ 3 の場合，初期解像度を低くしたことでオリジナルの解像度での最適解の離散直線の点群を含む離散直線が初期解像度での最適解から離れてしまうために，計算時間があまり削減できなかったと考えられる．具体的に述べると初期解像度を低くしすぎたためにオリジナルでは外れ値と見なされるものが解像度を落としたことで，低解像度においては離散直線を構成する点，すなわち内部点だと認識される解像度までは外れ値の集合が内部点の多い離散直線だとされるものが大量に発生する．それが本来の最適解より内部点が多い状態が続き，最適解を含む問題が解かれず，オリジナルの解像度での外れ値で構成された問題が優先的に解かれていく．しかし，ある解像度になると外れ値の集合であることや細かな離散直線の集合だと判断される．それにより次に内部点の多い問題を考えるがその問題も同様の問題であるということが繰り返され，最適解を含む問題が優先的に解かれないという状況になっていると考えられる．いいかえると枝刈りが起こらず，最悪の場合の計算量に近くなっていると考えられるということである．これは 4.4 節で考察した枝刈りがうまく機能しなくなる例だといえる．



(a) オリジナルの解像度でのデータ

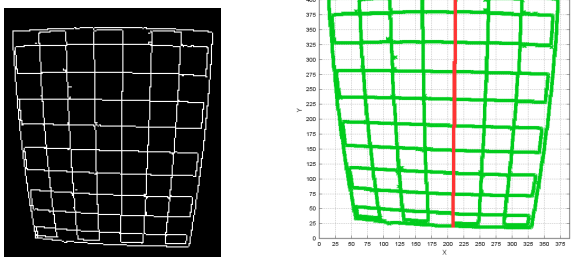


(b) オリジナルの解像度での提案手法における最適解

図 11　オリジナルの解像度での離散直線当てはめ
**Fig. 11**　Digital line fitting in original resolution.

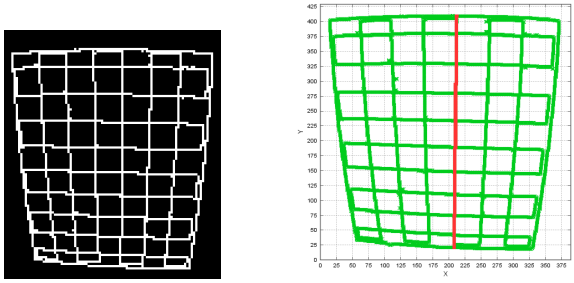
また，並列化処理の効果はデータにより異なる．これは，データに含まれる離散直線が一方の離散直線モデルのみを満たす場合と 2 つのモデルを均等に満たす場合，すなわち，データに含まれる離散直線の傾きが偏っている場合とデータに様々な傾きを持つ離散直線が含まれる場合で並列処理による計算時間の短縮が変化するからである．前者のようなデータ場合，計算時間の短縮は小さく，後者のようなデータの場合計算時間の短縮は大きくなると考えられる．

また，本手法はどのような初期解像度から探索を始めてもオリジナルの解像度で解いたものと同様の最適解が得られるが，このことを実験により確認するために，初期解像度を変更して離散直線当てはめを行い，すべての解像度で同様の最適解が得られることを示す．オリジナルの解像度のデータと解いた結果を**図 11**，初期解像度 1/2 でのデータと解いた結果を**図 12**，初期解像度 1/4 でのデータと解いた結果を**図 13**，初期解像度 1/8 でのデータと解いた結果を**図 14**，初期解像度 1/16 でのデータと解いた結果を**図 15**，初期解像度 1/32 でのデータと解いた結果を**図 16** にそれぞれ示す．なお，ここで示す結果はデータ 2 を用いたものである．結果の図では緑色の点が外れ値とされた



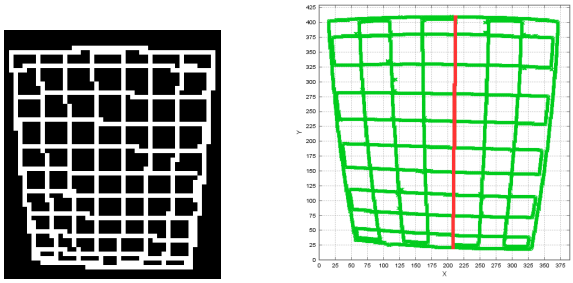
(a) 初期解像度 1/2 でのデータ　(b) 初期解像度 1/2 の場合での提案手法における最適解

図 12　初期解像度 1/2 での離散直線当てはめ
**Fig. 12**　Digital line fitting in 1/2 first resolution.



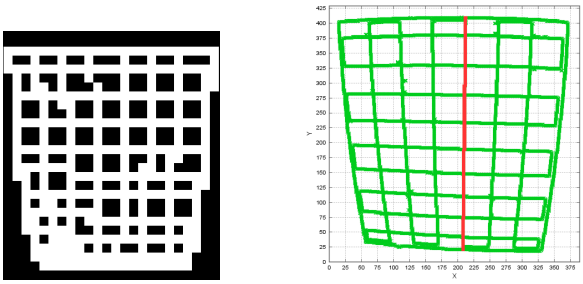
(a) 初期解像度 1/4 でのデータ　(b) 初期解像度 1/4 の場合での提案手法における最適解

図 13　初期解像度 1/4 での離散直線当てはめ
**Fig. 13**　Digital line fitting in 1/4 first resolution.



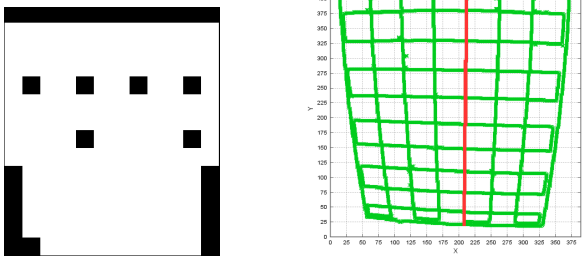
(a) 初期解像度 1/8 でのデータ　(b) 初期解像度 1/8 の場合での提案手法における最適解

図 14　初期解像度 1/8 での離散直線当てはめ
**Fig. 14**　Digital line fitting in 1/8 first resolution.



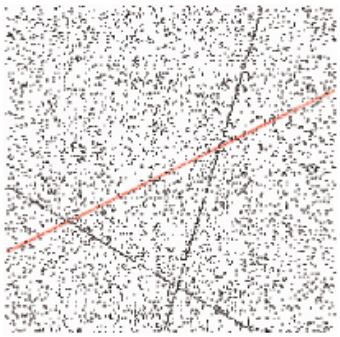
(a) 初期解像度 1/16 でのデータ　(b) 初期解像度 1/16 の場合での提案手法における最適解

図 15　初期解像度 1/16 での離散直線当てはめ
**Fig. 15**　Digital line fitting in 1/16 first resolution.

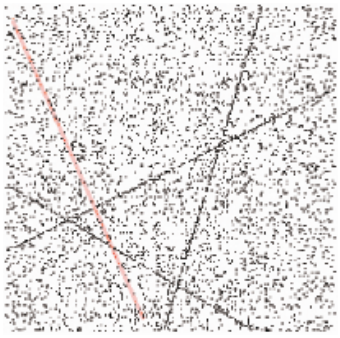


(a) 初期解像度 1/32 でのデータ　(b) 初期解像度 1/32 の場合での提案手法における最適解

図 16　初期解像度 1/32 での離散直線当てはめ
**Fig. 16**　Digital line fitting in 1/32 first resolution.



(a) 提案手法による直線当てはめ



(b) RANSAC による直線当てはめ

図 17　提案手法と RANSAC により得られる解の比較
**Fig. 17**　Comparing result by the proposed method with result by RANSAC.

点，赤色の点が最適解とされたものである．

結果を比較するとすべての場合で同じ離散直線が最適解として求まっていることが分かる．ここでは初期解像度 1/32 までの結果を示したが，解像度 1/32 でのデータを示す図 16 (a) は解像度が十分に低く，非常に単純な点の組合せになり，同一の最適解が得られることを確認している．

次に，提案手法により得られた離散直線の最適解と，従来非常によく用いられている RANSAC により得られる解を比較した結果を**図 17** に示す．同一の点群に対し直線当てはめを行い，それぞれで得られた解を赤い線で示している．これは，$102 \times 102$ の画像に 3 直線が含まれており，さらにランダムに 2,000 点の外れ値を生成したものである．提案手法では，内部点の数が 89 となる最適解が得られた

のに対し，RANSAC では 38,427 回サンプリングを行い 37 点の内部点しか得られなかった．RANSAC は最適解を得られることを保証していないため，最適解が得られるとは限らない．図 17 でも，提案手法では最適解が得られているのに対し，RANSAC では明らかに間違った解が得られていることが分かる．

さらに提案手法の RANSAC に対する優位性を示すために，対象の断面を得ることのできる距離センサで平面を計測した場合に得られるデータを想定した合成データに対して離散直線当てはめをそれぞれ行った．用いたデータを**図 18** に示す．

距離センサは，レーザなどを対象に照射して戻ってくるまでの時間や位相差により対象までの距離を得るセンサであり，ロボットなどに搭載されて環境認識に用いられている．一般に，センサは横方向に微小角ずつ回転しながら対象までの距離を得ることにより，対象のある高さでの断面を得る．本実験では，対象が平面であることを仮定し，距離センサを回転させながらデータを得たことを想定したデータを合成した．具体的には，原点にあるレンジカメラから $y$ 軸方向に 1,000 離れた離散直線のデータを得ると想定し，角度を 0.05° ずつ動かしながら ±85° までの範囲のデータを得て，各データはノイズには正規分布のノイズを付加した．このとき，(a) のデータはすべてのデータにノイズが付加されているが，(b) の一部のデータには意図的にノイズを付加していない．(a) は離散直線を構成する点が 3,327 点のみからなる．(b) はデータ点の総数は 3,451 点であり，そのうち対象の離散直線を構成する点が 3,327 点，外れ値は 124 点である．

まず，図 18 に対し，提案手法で離散直線当てはめを行った結果を**図 19** に示す．

図 19 では最適解に含まれる点を赤で，外れ値を緑で示している．計算時間は (a) については 434.072 s，(b) については 259.974 s であり，いずれも解の点数は 3,327 点であった．これは離散直線上にある点と一致し，提案手法で最適解が得られることを確認した．

次に図 18 に対し，RANSAC で離散直線当てはめを行った．提案手法が最適解を得るためにかかった計算時間，すなわち，(a) については 434.072 s，(b) については 259.974 s で求めることのできる最良の解（内部点が最も多くなる解）

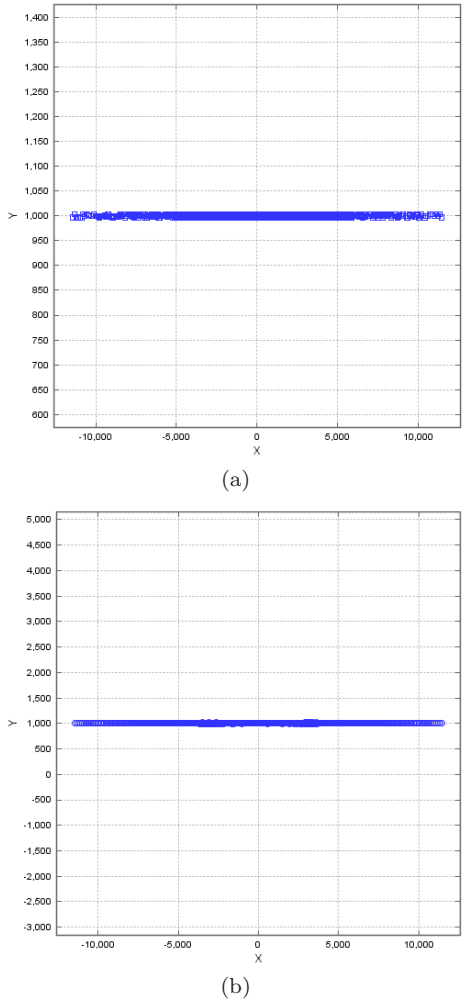



**図 18**　距離センサでのデータを想定した合成データ

**Fig. 18**　Synthetic data which assumes to be obtained using a range sensor.

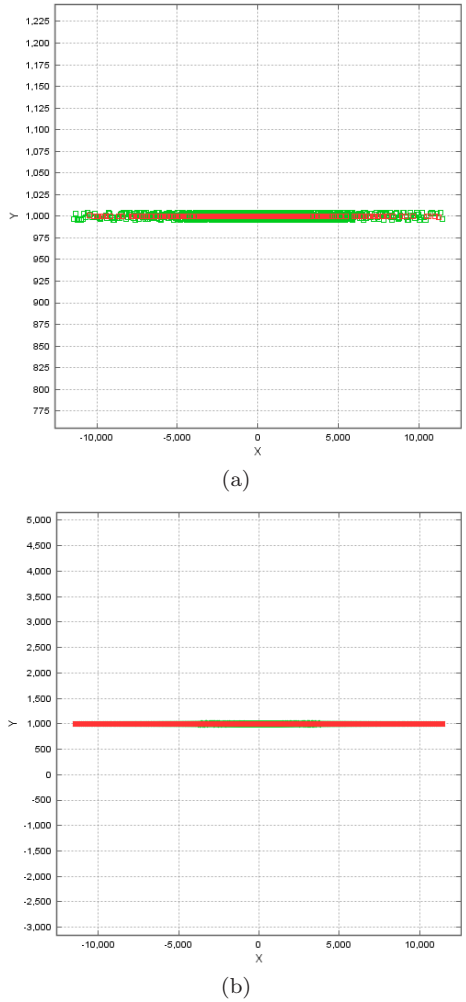
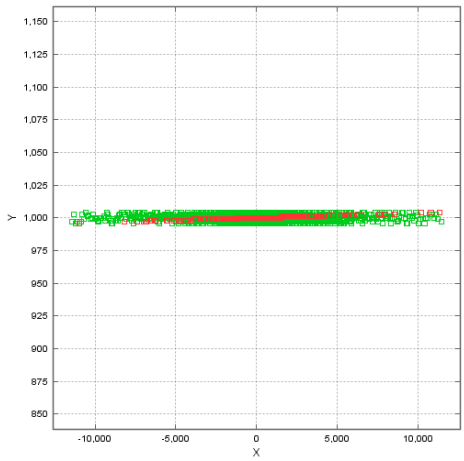
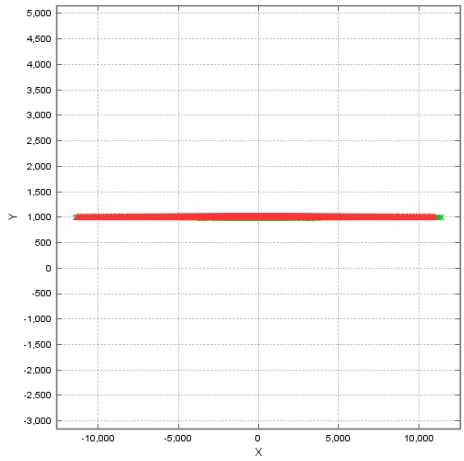



**図 19**　提案手法による離散直線当てはめの結果

**Fig. 19**　Result of digital line fitting of the proposed method.




(a)




(b)

**図 20**　RANSAC による離散直線当てはめの結果
**Fig. 20**　Result of digital line fitting of RANSAC.

**表 1**　RANSAC による離散直線当てはめの結果詳細
**Table 1**　Detail of result of digital line fitting of RANSAC.

| | (a) | | (b) | |
|---|---|---|---|---|
| 番号 | 内部点数 | サンプリング回数 | 内部点数 | サンプリング回数 |
| 1 | 547 | 2,397,376 | 3,325 | 1,435,413 |
| 2 | 550 | 2,275,679 | 3,326 | 1,433,241 |
| 3 | 547 | 2,282,199 | 3,325 | 1,423,118 |
| 4 | 542 | 2,134,942 | 3,325 | 1,437,486 |
| 5 | 547 | 2,143,109 | 3,322 | 1,442,066 |
| 6 | 547 | 2,128,050 | 3,325 | 1,437,241 |
| 7 | 528 | 2,152,132 | 3,326 | 1,432,967 |
| 8 | 532 | 2,268,154 | 3,326 | 1,450,110 |
| 9 | 550 | 2,281,543 | 3,325 | 1,441,417 |
| 10 | 547 | 2,419,799 | 3,326 | 1,445,313 |

を**図 20** に示す．RANSAC では，サンプリングされた 2 点が通る直線から −0.5 と 0.5 の間に入った点を内部点であるとする．なお，図 20 には RANSAC を 10 回実行した中で最も内部点の数が多かった解を示している．解として得られた内部点の数，RANSAC がサンプリングを行った回数を**表 1** に示す．

**表 2**　RANSAC で最適解を求めるための計算時間
**Table 2**　Computation time of calculate optimal solution by RANSAC.

| 番号 | 計算時間（s） | サンプリング回数 |
|---|---|---|
| 1 | 2,304.29 | 11,755,373 |
| 2 | 193.595 | 983,899 |
| 3 | 594.498 | 2,614,743 |
| 4 | 1,507.91 | 7,493,835 |
| 5 | 708.013 | 3,129,123 |
| 6 | 3,322.04 | 17,864,635 |
| 7 | 1,720.75 | 8,961,440 |
| 8 | 945.351 | 4,624,341 |
| 9 | 892.683 | 4,451,872 |
| 10 | 1,348.51 | 7,056,939 |
| 平均 | 1,353.76 | 6,893,620 |

(a) については，点数 550 程度が内部点に含まれる解しか得られず，(b) についてはすべての試行で最適解に近い解が得られているが，点数 3,327 の最適解は得られなかった．図 20 (a) および (b) でもその様子が見て取れる．

次に，(b) に対して RANSAC で最適解を得るために必要な時間を計測した結果を**表 2** に示す．実行回数は 10 回である．なお，(a) についてはすべての 2 点の組合せを探索しても最適解が得られなかった．(b) の一部では提案手法と比べ短時間で最適解が求められたものもあるが，多くの場合，RANSAC では提案手法に比べ最適解を求めるために多くの計算時間が必要であることが分かる．一方，提案手法は点の数の増加にかかわらず，決定的につねに同一の計算時間で最適解を求めることができる．これより提案手法は最適解を求めるうえでは RANSAC と比較して優位性があるといえる．さらに (a) では，すべての点が正規分布に従うノイズを含んでいるため，サンプリングされた点の微小な位置のずれにより幅を持った直線も最適解からずれたものになってしまう．これに対し提案手法では，すべての点がノイズを含んでいても，ノイズの大きささえ分かれば適切な直線の幅を設定することにより最適解を得ることができる．

最後に，直線や曲線などを含む実画像に対して直線当てはめを行った結果を示す．**図 21** の画像のエッジを検出し，細線化を施したものを入力とした．エッジ点の数は 1,746 である．これに離散直線当てはめをしたところ，オリジナルの解像度で解いたもの，多重解像度表現を用いて解いたものともに，最適解である点の数 520 の離散直線が得られた（**図 22**）．この図 22 では赤で示した点が外れ値，緑で示した点が最適解の離散直線に含まれる内部点である．

初期解像度 1/8 における計算過程を**表 3** に示す．また，その計算時間を**表 4** に示す．初期解像度を下げるにつれ，計算時間が短縮されていることが分かる．

表 3　2 次元における実データに対する離散直線当てはめの計算過程
**Table 3**　Calculation process of 2-D digital line fitting for real data.

| 計算回数 | 解像度 | 場合分け*1 | 入力の点の数*3 | 直線の数*2 | 最適解の点の数*3 |
|---|---|---|---|---|---|
| 1 | 1/8 | $x$ | 1,746 | 430 | 526 |
| 2 | 1/4 | $x$ | 534 | 292 | 522 |
| 3 | 1/2 | $x$ | 534 | 566 | 522 |
| 4 | 1/1 | $x$ | 534 | 1,068 | 520 |
| 5 | 1/8 | $y$ | 1,746 | 435 | 470 |
| 6 | 1/4 | $y$ | 526 | 260 | 466 |
| 7 | 1/2 | $y$ | 526 | 512 | 464 |
| 8 | 1/1 | $y$ | 526 | 992 | 463 |

*1　$(a \geq 0):$ と $(a < 0):$
*2　双対空間におけるトポロジカルスイープの直線の数
*3　オリジナルの解像度相当

図 21　2 次元における離散直線当てはめに使用した画像
**Fig. 21**　An image for 2-D digital line fitting.

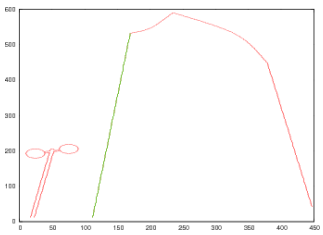


図 22　2 次元における離散直線当てはめの結果
**Fig. 22**　Result for 2-D digital line fitting.

表 4　2 次元における実データに対する離散直線当てはめの計算時間の比較
**Table 4**　Comparison of the calculation time between 2-D digital line fitting for real data.

| 初期解像度 | 計算時間 |
|---|---|
| 1/1 | 71.991 s |
| 1/2 | 39.102 s |
| 1/4 | 24.639 s |
| 1/8 | 20.635 s |

## 6.　おわりに

本論文では最適性を保証した多重解像度表現を用いた離散直線当てはめを提案した．離散直線当てはめの計算量は入力点の数が多くなればなるほど増大する．そこで，低い解像度で離散直線当てはめを行い，あらかじめ明らかに外れ値である点を除外することで，入力となる点の数を減らして離散直線当てはめを行うことで，計算時間を削減した．このとき，高い解像度で得られる離散直線はより低い解像度で得られる離散直線に必ず含まれるようなモデルを適用することにより最適性を保証した．具体的には，4 連結と呼ばれる離散直線モデルとダウンサンプリングの手法を適用した．さらに，問題の枝刈りや，似たような問題の統合により計算時間を削減した．また，2 つのモデルにより離散直線当てはめを並列化した．実験により，提案手法は多重解像度表現を用いない手法に比べて，計算時間を短縮していることを確認した．

今後の展望としては，今回提案した手法を 3 次元離散平面当てはめに応用することがあげられる．

**謝辞**　本研究の一部は，科学研究費補助金（No.24500198），栢森情報科学技術振興財団研究助成によった．

### 参考文献

[1] Kanatani, K.: *Statistical Optimization for Geometric Computation: Theory and Practice*, Elsevier Science Inc. (1996).
[2] Huber, P.J.: Finite sample breakdown of M- and P-estimators, *The Annals of Statistics*, Vol.12, pp.119–126 (1996).
[3] Fischler, M.A. and Bolles, R.C.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, *Comm. ACM*, Vol.24, Issue 6 (1981).
[4] Coeurjolly, D. and Brimkov, V.: Computational aspects of Digital Plane and Hyperplane Recognition, Lecture Notes in Computer Science, Vol.4040, pp.291–306 (2006).
[5] Stojmenović, I. and Tošić, R.: Digitaization schemes and the recognition of digital straigh lines, hyperplanes and flats in arbitrary dimensions, *Vision Geometry, Contemporary Mathematics Series*, Vol.119, pp.197–212 (1991).
[6] Gérard, Y.: A fast and elementary algorithm for digital plane recognition, *Electronic Notes in Discrete Mathematics*, Vol.12, pp.142–153 (2003).
[7] Gérard, Y., Debled-Renesson, I. and Zimmermann, P.:

An elementary digital plane recognition algorithm, *Discrete Applied Mathematics and Combinatorial Operations Research and Computer Science*, Vol.151, pp.169–183 (2005).
[8] Debled-Rennesson, I., Remy, J.-L. and Rouyer-Degli, J.; Linear segmentation of discrete curves into blurred segments, *Discrete Applied Mathematics*, Vol.151, pp.122–137 (2005).
[9] Provot, L., Buzer, L. and Debled-Rennesson, I.: Recognition of blurred pieces of discrete planes, *Discrete Geometry for Computer Imagery*, Lecture Notes in Computer Science, Vol.4245, pp.65–76 (2006).
[10] Kenmochi, Y., Buzer, L. and Talbot, H.: Efficiently computing optimal consensus of digital line fitting, *ICPR 2010*, pp.1064–1067 (2010).
[11] Zrour, R., Kenmochi, Y., Talbot, H., Shimizu, I. and Sugimoto, A.: Combinatorial optimization for robust digital line and plane detection, *WCVIM of PSIVT09* (2009).
[12] Churiki, M., Shimizu, I., Zrour, R., Talbot, H., Kenmochi, Y. and Sugimoto, A.: Digital line and plane recognition using combinatorial optimization by multiresolutional representation, *IEVC 2010* (2010).
[13] Said, M., Lachaud, J.-O. and Feschet, F.: Multiscale Discrete Geometry, *DGCI 2009*, LNCS, Vol.5810, pp.118–131 (2009).
[14] Lindeberg, T.: Scale-space theory: A basic tool for analysing structures at different scales, *J. of Applied Statistics*, Vol.21, No.2, pp.224–270 (1994).
[15] Lindeberg, T.: *Scale-Space Theory in Computer Vision*, Kluwer Academic Publishers, Dordrecht, Netherlands (1994).
[16] OpenCV, available from ⟨http://sourceforge.net/projects/opencvlibrary/⟩.

**宮武 孝尚**

2012 年東京農工大学工学部情報工学科卒業．2012 年同大学大学院工学府情報工学専攻入学．

**中力 雅人**

2009 年東京農工大学工学部情報コミュニケーション工学科卒業．2011 年同大学大学院工学研究科情報工学専攻博士前期課程修了．同年より野村総合研究所勤務．

**清水 郁子**　（正会員）

1994 年東京大学工学部計数工学科卒業．1999 年同大学大学院工学系研究科計数工学専攻博士課程修了．博士（工学）．同年埼玉大学工学部助手，2004 年東京農工大学大学院工学研究院講師，現在に至る．コンピュータビジョン，3 次元画像計測に興味を持つ．計測自動制御学会，IEEE-CS 等会員．