論　文

高速2回走査ラベル付けアルゴリズム

何　立風[†]　巣　宇燕[††]　鈴木　賢治[†††]　中村　剛士[††††]
伊藤　英則[††††]

An Efficient Two-Scan Connected-Component Labeling Algorithm

Lifeng HE[†], Yuyan CHAO[††], Kenji SUZUKI[†††], Tsuyoshi NAKAMURA[††††], and Hidenori ITOH[††††]

あらまし　本論文では，高速2回走査ラベル付けアルゴリズムを提案する．第一走査ではまず，対象物画素に暫定ラベルを付与する．走査中の各時点までに発見した同一対象物に属する暫定ラベルをその対象物に対応する同等ラベル集合にまとめ，それら暫定ラベルの代表ラベルを代表ラベルテーブルに記録する．二つの同等ラベル集合の間に連結関係を発見するたびに，その二つの同等ラベル集合を合併するとともに代表ラベルテーブルを更新する．第一走査後，同一対象物に属するすべての暫定ラベルが同一同等ラベル集合にまとめられ，同一代表ラベルをもつ．第二走査では，対象物画素に付与されていた暫定ラベルをその暫定ラベルの代表ラベルに書き換える．本アルゴリズムは原理的に単純で実装が容易という特徴をもつ．様々な性質の画像を用いた従来手法との比較評価実験により，本アルゴリズムが最速であることを示した．

キーワード　ラベル付け，ラスタ走査，連結成分，パターン認識

1. まえがき

パターン認識やコンピュータビジョンにおいて，2値画像中の対象物（連結成分）に別々のラベルを付けるラベル付けは，最も基本的かつ重要な処理の一つであり，コンピュータに対象物を認識させる上で欠くことができない[1]．ラベル付けは，エッジ検出やノイズ平滑化などの他の基本的画像処理手法と比べて演算時間が長いため，これらを組み合わせて構成されるパターン認識システムにおいて，そのオンライン実時間実現を妨げる決定的な要因となっていた．これはパターン認識システムにおけるラベル付け手法のボトルネック問題と呼ばれている．これまでに多くのラベル付けアルゴリズムが提案されており，次の4種類が代表的である．

(A)　マルチ走査アルゴリズム[3],[4]．これらのアルゴリズムは逐次局所処理により図形上でラベルを伝搬させ，画像の走査を複数回行うことによりラベル付けを行う．$N \times N$ 画素の画像では，最悪の場合に $N-2$ 回の走査が必要である．

(B)　2回走査アルゴリズム[2],[5]〜[9]．これらのアルゴリズムはラベルの連結性を記憶する一次元若しくは二次元テーブルを利用し，画像走査中または走査後にテーブル解析を行うことにより，2回の画像走査によりラベル付けを行う．複雑なテーブル解析が一般的に必要である．

(C)　ハイブリッドアルゴリズム[14]．このアルゴリズムは分類(A)と(B)のアルゴリズムの利点を組み合わせたものである．分類(A)と(B)のアルゴリズムより高速であり，最大4回の走査で完全なラベル付けを完了することが実験的に確認されている[13],[14]．

(D)　輪郭追跡アルゴリズム[13]．このアルゴリズムは走査中に発見した対象物の輪郭を追跡することによりラベル付けを行う．画像の走査が不規則となるため，並列処理，ハードウェア実装，オンライン実時間処理に向かない．

また，画像走査方法の違いから，分類(A)，(B)と

† 愛知県立大学情報科学研究科，愛知県
Graduate School of Information Science and Technology, Aichi Prefectural University, Aichi-ken, 480–1198 Japan
†† 名古屋産業大学環境情報マネジメント研究科，尾張旭市
Graduate School of Enviornment Management, Nagoya Sangyo University, Owariasahi-shi, 488–8711 Japan
††† シカゴ大学放射線医科学研究科，米国
Department of Radiology, The University of Chicago, USA.
†††† 名古屋工業大学工学研究科，名古屋市
Graduate School of Engineering, Nagoya Institute of Technology, Nagoya-shi, 466–8555 Japan

(C) はラスタ走査型，(D) は輪郭追跡型と大別することもある．

本論文では，新しい 2 回走査ラベル付けアルゴリズムを提案する．本アルゴリズムでは，同等ラベル集合と代表ラベルテーブルを用いて暫定ラベル間の連結性の解析を行うことにより高速なラベル付けを実現する．本アルゴリズムはラスタ走査型である分類 (A)，(B)，(C) 中最速のハイブリッドアルゴリズムに比べ，走査回数が少なく，連結性解析も効率的である．また，輪郭追跡アルゴリズムより，原理的に単純であり実装しやすいと考える．様々な性質の画像を用いた比較評価実験により，本アルゴリズムの有効性を示す．

2. 提案アルゴリズムの概要

2.1 第一走査

$N \times N$ 画素の 2 値画像中の対象物画素の値を V_O，背景画素の値を V_B とする．また，(x, y) $(0 \leq x \leq N-1, 0 \leq y \leq N-1)$ にある画素を $b(x, y)$，その画素の値を $v(x, y)$ で表す．ほとんどのラスタ走査ラベル付けアルゴリズムと同様，図 1 に示すマスクを用いてラスタ走査の順序で走査しながら，$b(x, y)$ に対して，次式の処理により暫定的なラベル付けを行う．本論文では，8 連結の連結成分のラベル付けアルゴリズムを示すが，4 連結への変更はマスク形状の変更などにより容易に行える．ただし，V_O, V_B（$V_O < V_B$）は任意の暫定ラベルよりも大きな値とし，暫定ラベル値 m は例えば 1 に初期化する．

$$v(x,y) = \begin{cases} \text{non operation} & \text{if } v(x,y) = V_B \\ m, (m = m+1) & \text{else if } v_{\min} = V_B \\ v_{\min} & \text{otherwise} \end{cases} \quad (1)$$

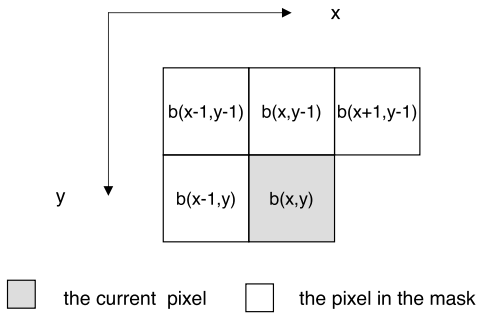


図 1　8 連結用走査マスク
Fig. 1　Mask for the eight-connected connectivity.

$$v_{\min} = \min[\{v(i,j) \mid \forall b(i,j) \in M_S\}] \quad (2)$$

ただし，$(m = m+1)$ は暫定ラベル値を 1 だけ増加させることを表し，$min(\cdot)$ は最小値を求める演算子，M_S はマスク中の四つの画素，つまり，$b(x-1, y)$，$b(x-1, y-1)$，$b(x, y-1)$ と $b(x-1, y+1)$ の集合，式中の条件は上の行を優先して判断する．

第一走査後，与えられた画像にある各対象物画素の値はある暫定ラベルとなる．一方，各背景画素の値は V_B のまま変わらない．

図 2 (a) に示す対象物に対して，ラスタ走査による対象物画素の処理順は a，b，c，…，l，m となる．対象物画素 a，b，c，g と j を処理する場合，それぞれの画素に対応するマスクに対象物画素が存在しないため，式 (1) の 2 行目より，注目画素に新たな暫定ラベルが付けられる．それ以外の対象物画素を処理するときは，それぞれの画素に対応するマスクに対象物画素が存在することから，式 (1) の 3 行目より，注目画素にマスク内の最小の暫定ラベルが付けられる．例えば，対象物画素 k を処理するとき（図 2 (b)），対応するマスクに暫定ラベル 5 と 2 が存在する．最小暫定ラベルは 2 であるから，画素 k に暫定ラベル 2 が付けられる．第一走査による暫定ラベル付けの結果は図 2 (c) のようになる．

2.2 連結性の解決

本アルゴリズムでは，同一対象物に属する暫定ラベルを同等ラベルと呼ぶ．走査中の各時点までに発見した同等ラベル l_1，…，l_n を同等ラベル集合と呼ぶ集合にまとめ，それらのラベル中の最小ラベル i を代表ラベルとする．便宜上，その同等ラベル集合を $\mathcal{S}(i) = \{l_1, \ldots, l_n\}$ で表す．また，暫定ラベルと代表ラベルの対応関係を代表ラベルテーブルと呼ぶテーブル $\mathcal{T}[\ ]$ に記録する．例えば，ある対象物に暫定ラベル 2，5，4，3 が付けられたとき，2 は代表ラベルであり，その対象物に対応する同等ラベル集合は $\mathcal{S}(2) = \{2, 5, 4, 3\}$ と表し，$k \in \{2, 5, 4, 3\}$ のような

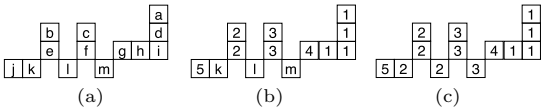


図 2　第一走査による暫定ラベル付け
Fig. 2　Illustration of provisional labeling by the first scan. (a) Connected-component example; (b) When processing object pixel k; (c) Provisional labels assigned after the first scan.

k において，$\mathcal{T}[k] = 2$ が成立する．

走査中，注目画素に暫定ラベル m が初めて付けられた場合，その時点で，対象物はその暫定ラベルしかもたないため，$\mathcal{S}(m) = \{m\}$，または $\mathcal{T}[m] = m$ になる．

一方，注目画素に既存暫定ラベル j が付けられた場合，$\mathcal{T}[j] = u$，つまり，暫定ラベル j の代表ラベルは u であるとする．これには次の 3 通りの場合が考えられる．(1) マスクに異なる暫定ラベルが存在しない場合．この場合は，何もしなくてもよい；(2) マスクに異なる暫定ラベル k が存在し，その暫定ラベルの代表ラベルも u である場合．暫定ラベル j と k が同じ代表ラベルをもつため，これらが同一同等ラベル集合に属していることが分かる．そのため，何もしなくてもよい；(3) マスクに u と異なる代表ラベル v をもつ暫定ラベル k が存在する，つまり，暫定ラベル j と k が別々の同等ラベル集合に属す場合．このとき，暫定ラベル j と k が同一対象物に属することから，暫定ラベル j と k の連結性を解決する必要がある．

上記の (3) の連結性は簡単に解決できる．暫定ラベル j と k が連結していることから，暫定ラベル j に対応する同等ラベル集合 $\mathcal{S}(u)$ と暫定ラベル k に対応する同等ラベル集合 $\mathcal{S}(v)$ に属するすべての暫定ラベルが同一対象物に属し，同等ラベルであることが分かる．そのため，同等ラベル集合 $\mathcal{S}(u)$ と $\mathcal{S}(v)$ を合併しなければならない．同等ラベル集合の定義により，$u < v$ のとき，$\mathcal{S}(v)$ を $\mathcal{S}(u)$ に合併し，$\mathcal{S}(v)$ に属するすべての暫定ラベル l に対して，$\mathcal{T}[l] = u$ にすればよい．一方，$u > v$ のとき，$\mathcal{S}(u)$ を $\mathcal{S}(v)$ に合併し，$\mathcal{S}(u)$ に属するすべての暫定ラベル l に対して，$\mathcal{T}[l] = v$ にする．

図 2 (a) の対象物において，対象物画素 p を処理する前のラベル付けの状況を図 3 (a) に示す．暫定ラベル 1 と 4 は同等ラベル集合 $\mathcal{S}(1)$ に属し，代表ラベル 1 をもつ．つまり，$\mathcal{S}(1) = \{1, 4\}$，$\mathcal{T}[1]$=1，$\mathcal{T}[4]$=1 が成立する．一方，暫定ラベル 2，3 と 5 は同等ラベル集合 $\mathcal{S}(2)$ に属し，代表ラベル 2 をもつ．つまり，$\mathcal{S}(2) = \{2, 3, 5\}$，$\mathcal{T}[2] = 2$，$\mathcal{T}[3] = 2$，と $\mathcal{T}[5] = 2$ が成立する．対象物画素 p を処理することによって，暫定ラベル 3 と 4 が連結していることが分かる．$\mathcal{T}[3] = 2$ と $\mathcal{T}[4] = 1$ であるから，暫定ラベル 3 と 4 はそれぞれ異なる同等ラベル集合 $\mathcal{S}(2)$ と $\mathcal{S}(1)$ に属するため，$\mathcal{S}(2)$ と $\mathcal{S}(1)$ を合併しなければならない．合併の結果，図 3 (b) に示すように，$\mathcal{S}(1)$=$\{1, 4, 2, 3, 5\}$ となり，暫定ラベル 2，3，5 の代表ラベル，つまり，$\mathcal{T}[2]$，$\mathcal{T}[3]$ と $\mathcal{T}[5]$ の値を同等ラベル集合 $\mathcal{S}(1)$ の代表ラベル 1 に書き換える．

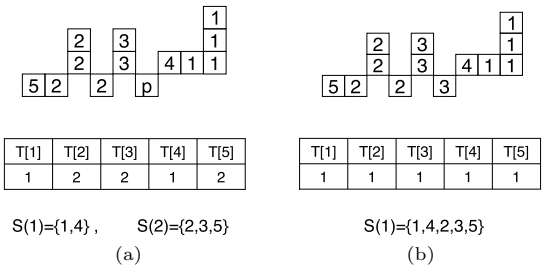


図 3 連結性の解決
Fig. 3 Temporary equivalent label sets and label-equivalence resolving. (a) Before processing of pixel p; (b) After processing of pixel p.

2.3 第二走査

本アルゴリズムでは，二つの同等ラベル集合の間に連結関係を発見するたびに，その二つの同等ラベル集合を合併するともに代表ラベルテーブルを更新することから，第一走査後，同一対象物に属するすべての暫定ラベルが同一同等ラベル集合にまとめられ，同一代表ラベルをもつことになる．そのため，第二走査では，対象物画素に付与された暫定ラベルをその暫定ラベルの代表ラベルに書き換えれば，同一対象物のすべて画素が同一ラベルをもつことになる．あらかじめ $\mathcal{T}[V_B] = 0$(注1)としておけば，そのラベルの書換えは次の演算で簡単に実現できる．

$$v(x, y) = \mathcal{T}[v(x, y)].$$

2.4 暫定ラベル付けの簡略化

本アルゴリズムでは，連結関係を発見するたびにそれを解決するため，注目画素の処理を終えると，マスク中のすべての暫定ラベルが必ず同一同等ラベル集合に属し，同一代表ラベルをもつ．そのため，マスクに対象物画像が存在する場合，式 (1) の 3 行目のように，注目画素にマスク中の最小の暫定ラベルを付与する必要がなく，マスクにある任意の暫定ラベルを付与することができる(注2)．これにより，マスク中の暫定ラベルの最小値の計算を避けられ，ラベル付けを効率化できる．

(注1)：つまり，第二走査では，背景画素の値を V_B から 0 に書き換える．
(注2)：この方法は他のラスタ走査ラベル付けアルゴリズムにも適用できると思われる．

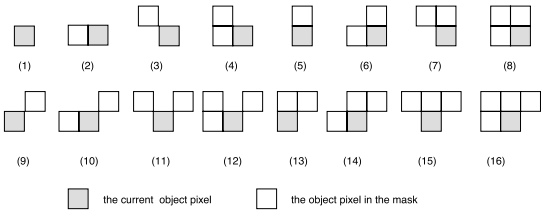


図 4　マスク中にとり得る 16 種類のケース
Fig. 4　Sixteen possible cases for the current object pixel in the mask for the eight-connected connectivity.

2.5　マスク中パターンのケース分析による効率化

注目画素が対象物画素であるとき，マスク中には図 4 に示す 16 種類のパターンをとり得る．

ケース (1) は注目画素に新たな暫定ラベルを付けなければならない唯一のケースである．

ケース (2)〜(9) 及び (13)〜(16) では，マスク中の暫定ラベル間の連結関係は考える必要がなく，注目画素にマスク中の任意の暫定ラベルを付ければよい．すなわち，これらのケースでは，マスク中に暫定ラベル間の連結関係が存在しても，その連結関係はその時点までの処理により既に解決済みとなる．例えば，ケース (16) を考えると，マスクにある対象物画素は $b(x-1,y-1) \rightarrow b(x,y-1) \rightarrow b(x+1,y-1) \rightarrow b(x-1,y)$ の順に処理される．$b(x,y-1)$ の処理後，$v(x,y-1)$ と $v(x-1,y-1)$ が同一同等ラベル集合に属し，同一代表ラベルをもつ．$v(x+1,y-1)$ の処理後，$v(x,y-1)$，$v(x-1,y-1)$ と $v(x+1,y-1)$ が同一同等ラベル集合に属し，同一代表ラベルをもつ．同様に，$b(x-1,y)$ の処理後，これら四つの画素にが同一同等ラベル集合に属し，同一代表ラベルをもつことになる．このように，これらのケースでは連結関係を考える必要がない．

ケース (10) と (11) では，もともとは独立な二つの対象物画素が注目画素を通して連結した場合である．この場合，その二つの画素に付けられた暫定ラベル間の連結関係が解決済みかどうかを調べる必要がある．もし，その二つの暫定ラベルが同一代表ラベルをもっていれば，その二つの暫定ラベル間の連結関係が既に解決済みであることを意味する．一方，その二つの暫定ラベル間の代表ラベルが異なれば，その二つの暫定ラベル間の連結関係を解決する必要がある．また，いずれの場合においても，注目画素にその二つの暫定ラベルのいずれかを付ければよい．

表 1　マスク中にとり得る 16 ケースにおける連結性解決の演算
Table 1　Operations in the 16 cases.

Case	b_4	b_3	b_2	b_1	$v(x,y)$	Operations for $\mathcal{T}[\,]$, $\mathcal{S}(\,)$ and m
(1)	0	0	0	0	m	$\mathcal{T}[m]=m$, $\mathcal{S}(m)=\{\mathrm{m}\}$, $m=m+1$
(2)	0	0	0	1	v_1	no operation
(3)	0	0	1	0	v_2	no operation
(4)	0	0	1	1	v_1 or v_2	no operation
(5)	0	1	0	0	v_3	no operation
(6)	0	1	0	1	v_1 or v_3	no operation
(7)	0	1	1	0	v_2 or v_3	no operation
(8)	0	1	1	1	v_1, v_2, or v_3	no operation
(9)	1	0	0	0	v_4	no operation
(10)	1	0	0	1	v_1 or v_4	$resolve(v_1,v_4)$
(11)	1	0	1	0	v_2 or v_4	$resolve(v_2,v_4)$
(12)	1	0	1	1	v_1, v_2, or v_4	$resolve(v_1,v_4)$ or $resolve(v_2,v_4)$
(13)	1	1	0	0	v_3 or v_4	no operation
(14)	1	1	0	1	v_1, v_3, or v_4	no operation
(15)	1	1	1	0	v_2, v_3, or v_4	no operation
(16)	1	1	1	1	v_1, v_2, v_3, or v_4	no operati on

ケース (12) では，対象物画素 $b(x-1,y)$ の処理後，$v(x-1,y)$ と $v(x-1,y-1)$ が同一同等ラベル集合に属し，同一代表ラベルをもつことになる．このため，$v(x-1,y)$ と $v(x-1,y-1)$ のどちらも連結性の解決に用いることができる．$v(x-1,y)$ または $v(x-1,y-1)$ を用いる場合，それぞれケース (10) またはケース (11) と同様に処理できる．

以上の議論を表 1 にまとめる．ただし，b_1，b_2，b_3 と b_4 をそれぞれ画素 $b(x-1,y)$，$b(x-1,y-1)$，$b(x,y-1)$ と $b(x+1,y-1)$ を表し，0 は背景画素，1 は対象物画素を示す．また，v_1，v_2，v_3 と v_4 をそれぞれ b_1，b_2，b_3 と b_4 の値を表す．更に，$resolve(v_i,v_j)$ は対象物画素 v_i の暫定ラベルと対象物画素 v_j の暫定ラベル間の連結関係を解決するための演算を表す．

二つの暫定ラベル間の連結関係を解決するための処理 *resolve* は複雑であるから，この演算は極力避けるべきである．表 1 の 16 ケースにおける演算 *resolve* に対応するカルノー図 [15] を図 5 に示す．これにより，演算 *resolve* が発生しない条件は次の論理式で表せる．

$$b_3 + \overline{b_4} + \overline{b_1} \cdot \overline{b_2} \tag{3}$$

ただし，b_i が対象物画素であるとき，b_i が真，$\overline{b_i}$ が偽である．また，$(\cdot)$ は論理積，$(+)$ は論理和を表す．

式 (3) により，連結関係を解決する演算 *resolve* を

b3 b4 \ b1 b2	0 0	0 1	1 1	1 0
0 0	0	0	0	0
0 1	0	1	1	1
1 1	0	0	0	0
1 0	0	0	0	0

図 5　演算 *resolve* における Karnaugh 図
Fig. 5　The Karnaugh map for the operation *resolve*.

行わない条件として，b_3 が対象物画素であるか，b_4 が背景画素であるか，または b_1 及び b_2 が背景画素であるかのいずれかである．暫定ラベルを付けなければならない画素は注目画素であることから，b_3 が対象物画素であるかどうかを最初にチェックするのが最も効率的である．もし，b_3 が対象物画素であれば，演算 *resolve* は必要なく，注目画素に b_3 に付けられた暫定ラベル v_3 を付けるだけでよい．一方，他の画素，例えば，b_4 を最初にチェックすることを考えると，もし，b_4 が対象物画素でなければ，演算 *resolve* は必要ないが，注目画素に暫定ラベルを付けるために，マスク中の他の画素を調べる必要がある．マスク中の各画素が対象物画素である確率が同じであるとすれば，後者の平均処理時間が前者より長いことは明らかである．

2.6　提案したアルゴリズムの要約

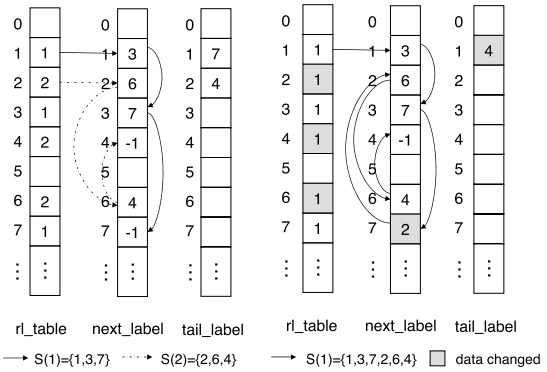
以上の議論により，提案したアルゴリズムは次のようにまとめられる．ただし，所用記号の意味は前述と同様である．

```
%% 第一走査 (First scan)
m = 1;
for (y = 0; y < N; y++)
    for (x = 0; x < N; x++)
        if (v(x,y) ≠ V_B)
            if (v_3 ≠ V_B)
                v(x,y) = v_3;
            else if (v_1 ≠ V_B)
                v(x,y) = v_1;
                if (v_4 ≠ V_B)
                    resolve(v_1, v_4);
                end if
            else if (v_2 ≠ V_B)
                v(x,y) = v_2;
                if (v_4 ≠ V_B)
                    resolve(v_2, v_4);
                end if
            else if (v_4 ≠ V_B)
                v(x,y) = v_4;
            else
                v(x,y) = m;
                T[m] = m;
                S(m) = {m};
                m = m + 1;
            end if
        end if
    end for
end for
%% 第二走査 (Second scan)
T[V_B] = 0;
for (y = 0; y < N; y++)
    for (x = 0; x < N; x++)
        v(x,y) = T[v(x,y)];
    end for
end for
```

3.　アルゴリズムの実装

代表ラベルテーブルは一次元配列 *rl_table*[ ] により簡単に実現できる．すなわち，暫定ラベル l の代表ラベルを r にするときは，*rl_table*[l]=r とすればよい．また，暫定ラベル t の代表ラベル s は s=*rl_table*[t] で調べられる．

同等ラベル集合に対する主な演算は二つの集合の合併である．同等ラベル集合を連結リストで実現すれば，その演算は定数回で終了する．便宜上，連結リストで実現した同等ラベル集合のことを同等ラベルリストと表現することにする．各暫定ラベルは一つの同等ラベルリストにしか属さないことから，同等ラベルリストは，*rl_table*[ ] と二つの一次元配列 *next_label*[ ] と *tail_label*[ ] で実現できる．*next_label*[ ] は同等ラベルリストにおいてある暫定ラベルの次の暫定ラベルを示す配列である．*next_label*[l] の値は j であるとき，暫定ラベル l の次のラベルは j であることを示す．ただし，*next_label*[t] の値は -1 であるとき，暫定ラベル t の次のラベルは存在しない，つまり，t は同等ラベルリストの末尾要素であることを表す．一方，*tail_label*[ ] は同等ラベルリストの末尾要素を示す配列である．*tail_label*[u] の値は v であることは，同等ラベルリスト $\mathcal{S}(u)$ の末尾要素が v であることを表す．



(a) before connecting (b) after connecting

図 6 連結性解決のための同等ラベルリストの合弁
Fig. 6 Operations for label equivalence resolving.

$rl\_table[\,]$, $next\_label[\,]$ と $tail\_label[\,]$ を用いれば，任意の暫定ラベル l が属する同等ラベルリストの先頭要素（つまり，l の代表ラベル）u は $u{=}rl\_table[l]$ により，そのリストの末尾要素 v は $v{=}tail\_label[rl\_table[l]]$ により求められる．これにより，同等ラベルリストの新設と合併は簡単に実現できる．同等ラベルリスト $\mathcal{S}(l) = \{l\}$ を新設するためには，次の操作を行えばよい．

$rl\_table[l]{=}l$, $next\_label[l]{=}-1$, $tail\_label[l]{=}l$.

一方，同等ラベルリスト $\mathcal{S}(v)$ を $\mathcal{S}(u)$ に合併するときは，$\mathcal{S}(v)$ の先頭を $\mathcal{S}(u)$ の末尾に連結させ，$\mathcal{S}(v)$ にあるすべての暫定ラベルの代表ラベルを u に書き換えればよい．これは次の操作により実現できる．

$next\_label[tail\_label[u]] = v;$
$tail\_label[u] = tail\_label[v];$
$i = v;$
while $(i \neq -1)$ do
  $rl\_table[i] = u;$
  $i = next\_label[i];$
end while

同等ラベルリストの合弁の例として，図 6 (a) に，同等ラベルリスト $\mathcal{S}(1) = \{1, 3, 7\}$ と $\mathcal{S}(2) = \{2, 6, 4\}$ に対応する $rl\_table[\,]$, $next\_label[\,]$ と $tail\_label[\,]$ を示す．$\mathcal{S}(2)$ を $\mathcal{S}(1)$ に合併した結果を図 6 (b) に示す．

なお，配列 $rl\_table[\,]$, $next\_label[\,]$ と $tail\_label[\,]$ のいずれのサイズも最大の暫定ラベルに依存する．$N \times N$ 画素の画像において，最大の暫定ラベルの数は $N \times N/4$ であるため，これらの配列のサイズを $N \times N/4$ にすればよい．

4. 比較評価実験

提案アルゴリズムの特性及び性能を評価するために，ワークステーション（Intel Pentium D Duo 3.0 GHz + 3.0 GHz CPUs，2 GByte Memory，Mandriva Linux OS）上での CPU 実行時間を用いた比較評価実験を行った．なお，実行時間は 5,000 回の実行の平均値である．

実験には次の 5 種類の画像を用いた．(1) ノイズ画像：五つのサイズ（32×32，64×64，128×128，256×256 と 512×512 画素）の画像に，1,000 階調までの白色一様ノイズを加え，しきい値を 0 から 1,000 まで 25 ごとに変化させて 2 値化した合計 205 種類の 2 値画像（各サイズごとには 41 種類）．このような画像は自然画像に比べ，複雑な幾何学形状と複雑なラベル連結性をもつため，ラベル付けアルゴリズムをより厳格に評価するのに適している．(2) 自然画像：SIDBA（Standard Image Data Base）及び USC（University of Southern California）の標準画像データベースから風景画像，航空画像，人物画像，静物画像，指紋画像，スナップ画像及びテキスト画像を含む 50 種類の自然画像を Otsu のしきい値選択法 [11] により 2 値化した画像．(3) 医用画像：シカゴ大学医用画像データベースから 25 種類の医用画像を Otsu 法 [11] により 2 値化した画像．(4) Columbia-Utrecht Reflectance and Texture Database (注3) から 7 種のテクスチャ画像を Otsu 法 [11] により 2 値化した画像．(5) テストパターン画像：ラベル付けアルゴリズムを評価するため作成した，階段状，渦状，のこぎり状，チェッカーボード状，蜂巣状パターンを含む人工画像 [14]．

ハイブリッドアルゴリズム [14] がラスタ走査型の分類 (A)，(B)，(C) 中最速であること [14]，輪郭追跡アルゴリズム [13] が最新のアルゴリズムであることから，この二つのアルゴリズムを比較対象とした．なお，実験に用いたハイブリッドアルゴリズムのプログラムは著者から提供されたものであり，輪郭追跡アルゴリズムのプログラムは著者のホームページ (注4) よりダウンロードしたものである．

図 7 に，ノイズ画像における画像サイズに対する各アルゴリズムの実行時間特性を示す．三つのアルゴリズムとも線形特性をもつことが確認できる．画像サ

(注3)：http://www1.cs.columbia.edu/CAVE/software/curet/index.php
(注4)：http://dar.iis.sinica.deu.tw/Download

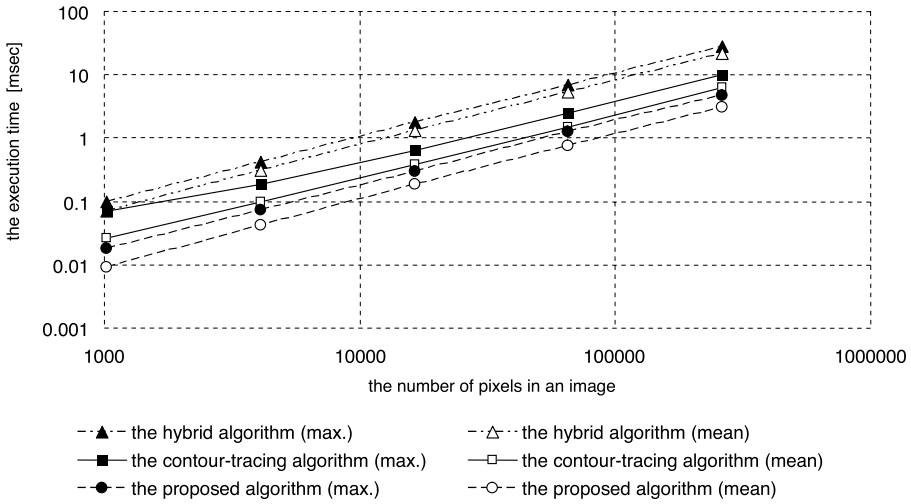


図 7　画像サイズに対する実行時間の線形性
Fig. 7　Linearity of the execution time versus image size.

表 2　自然画像，医用画像，テクスチャ画像，テストパターン画像における実験結果（ミリ秒）
Table 2　Comparison of various execution times [ms] on natural images, medical images, textural images, and special-shape artificial images.

Image type		Hybrid	Contour	Ours
natural	max.	18.4	3.8	**2.3**
	mean	13.7	2.4	**1.5**
	min.	9.6	1.2	**1.0**
medical	max.	14.9	2.6	**1.5**
	mean	13.4	1.9	**1.2**
	min.	11.4	1.5	**0.9**
textural	max.	28.5	3.7	**2.0**
	mean	27.4	2.7	**1.6**
	min.	26.4	1.5	**1.1**
artificial	max.	15.8	7.4	**1.8**
	mean	7.9	3.9	**0.8**
	min.	2.2	1.1	**0.3**

表 3　他の基本的な画像処理手法との実行時間（ミリ秒）の比較
Table 3　Comparison of execution time [ms] with other fundamental image-processing methods.

Algorithm	Execution time [ms]
our algorithm (max.)	**2.3**
our algorithm (mean)	**1.5**
thresholding	4.6
edge detection	9.6
noise reduction	51.5

イズに対する線形性は，パターン認識アルゴリズムにとって望ましい特性である．なお，これら三つ以外のラベル付けアルゴリズムは，線形特性をもっていない [14]．また，本アルゴリズムの最大実行時間が他の二つの手法の平均実行時間よりも短いことも確認できる．

自然画像，医用画像，テクスチャ画像と人工画像を用いた実験結果を表 2 に示す．すべての種類の画像において，本アルゴリズムが最速であることが分かる．

また，表 3 は，自然画像，医学画像，テクスチャ画像を用いて，2 値化処理（しきい値選択を含む）[11]，エッジ検出（Marr-Hildreth の演算子）[10]，ノイズ平滑化（3×3 画素マスクの中央値フィルタ）[12] など基本的な画像処理手法の実行時間と，本アルゴリズムの実行時間を比較したものである．本アルゴリズムによるラベル付けの実行時間が，最も短いことを確認できる．

5. 考　　察

本章は提案したアルゴリズムの効率化の要因について分析する．まず，注目画素が背景画素であるときには，何も処理をしないことから，効率化に関与しない．そのため，注目画素が対象物画素であるときだけ分析すればよい．また，便宜上，マスクの図 4 に示すマスク中の 16 種類のパターンが同確率での現れることとする．

提案したアルゴリズムの効率化の要因には，次の三つがあると考えられる．(1) **2. 4** に示した暫定ラベル

付けの簡略化（効率手法 1 と呼ぶ）; (2) **2.5** に示したマスク中パターンのケース分析による b_3 から対象物画素であるかどうかをチェックする方法（効率手法 2 と呼ぶ）; (3) 提案したアルゴリズムの特有の連結性の解決手法を実現したデータ構造（効率手法 3 と呼ぶ）.

従来のラスタ走査手法では，注目対象物画素にマスク中の最小の暫定ラベルを付ける．この最小の暫定ラベルを求めるため，どのケースにおいても 3 回の比較が必要である．それに対して，提案したアルゴリズムの場合，その比較回数がケース (5)，(6)，(7)，(8)，(13)，(14)，(15) 及び (16) のとき 1 回，ケース (2)，(4)，(10) 及び (12) のとき 2 回，ケース (3)，(11) 及び (15) のとき 3 回，ケース (1) 及び (9) のとき 4 回となる．その平均比較回数は $(1\times8+2\times4+3\times3+4\times2)/16 = 33/16 \approx 2.06$ 回となり，従来手法の 3 回より少ない．

一方，もし b_3 からではなく，$b_1 \rightarrow b_2 \rightarrow b_3 \rightarrow b_4$ の順で b_i は対象物画素であるかどうかをチェックする場合，そのチェック回数はケース (5)，(6)，(7)，(8)，(13)，(14)，(15) 及び (16) のとき 3 回，ケース (1)，(2)，(3)，(4)，(9)，(10)，(11) 及び (12) のとき 4 回となる．その平均回数は $(3\times8+4\times8)/16 = 3.5$ 回である．これに対し，提案手法のように，$b_3 \rightarrow b_1 \rightarrow b_2 \rightarrow b_4$ の順でチェックする場合，そのチェック回数はケース (5)，(6)，(7)，(8)，(13)，(14)，(15) 及び (16) のとき 1 回，ケース (2)，(4)，(10) 及び (12) のとき 3 回，ケース (1)，(3)，(9) 及び (10) のとき 4 回となる．その平均回数は $(1\times8+3\times4+4\times4)/16=2.25$ 回となり，従来手法の 3.5 回より少ない．

また，効率手法 1 と効率手法 2 以外の効率化部分は効率手法 3 によるものと考えられる．

4. に示したノイズ画像を用いて効率化手法の分析実験を行った結果を図 8 に示す．ただし，algorithm A，algorithm B，algorithm C と algorithm D は，それぞれ効率手法 1 と効率手法 2 の取込みなし，効率手法 1 の取込みなし，効率手法 2 の取込みなし，及び提案したアルゴリズムを表す．

6. む す び

本論文では，高速 2 回走査ラベル付けアルゴリズムを提案した．様々な性質の画像を用いた比較評価実験により，提案したアルゴリズムが最速であることを示した．本アルゴリズムにより，パターン認識分野におけるラベル付け手法のボトルネック問題を解決できると考えられる．

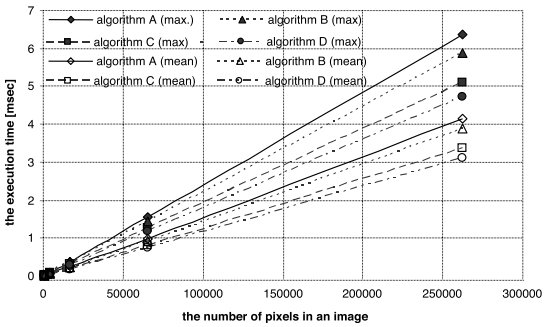


図 8　効率化手法の分析
Fig. 8　The analysis of efficient methods.

今後の研究として，提案したアルゴリズムの並列化や三次元画像のラベル付けへの拡張が挙げられる．

謝辞　本研究の一部は愛知県立大学海外研究の支援及び豊秋奨学会の研究助成によりアメリカシカゴ大学で完成したものである．また，貴重な御指摘とコメントを頂いた査読者に感謝する．

文　　献

[1] 鳥脇純一郎，画像理解のためのディジタル画像処理［II］, pp.45–47, 昭晃堂，東京，1988.

[2] A. Rosenfeld and J.L. Pfalts, “Sequential operations in digital picture processing,” J. ACM, vol.13, no.4, pp.471–494, Oct. 1966.

[3] R.M. Haralick, “Some neighborhood operations,” in Real Time/Parallel Computing Image Analysis, pp.11–35, Plenum Press, New York, 1981.

[4] 橋詰明英，鈴木隆一，横内久猛，堀内秀之，山本真司，“赤血球自動識別アルゴリズムとその評価,” 医用電子と生体工学，vol.28, no.1, pp.25–32, Jan. 1990.

[5] R. Lumia, L. Shapiro, and O. Zungia, “A new connected components algorithm for virtual memory computers,” Comput. Vis., Graph. Image Process., vol.22, no.2, pp.287–300, Feb. 1983.

[6] Y. Shirai, “Labeling connected regions,” in Three-Dimensional Computer Vision, pp.86–89, Springer-Verlag, New York, 1987.

[7] 後藤敏行，太田善之，吉田真澄，白井良明，“連結領域の高速ラベル付けアルゴリズム,” 信学論（D-II），vol.J72-D-II, no.2, pp.247–255, Feb. 1989.

[8] 奥山良幸，小林芳樹，武長　寛，浅田和佳，藤原和紀，“ラスタ走査形ラベリングの高速化手法,” 信学論（D-II），vol.J73-D-II, no.1, pp.36–45, Jan. 1990.

[9] 直井　聡，“文字の形状特徴を利用した可変ウィンドウサイズによる高速ラベリング手法,” 信学論（D-II），vol.J80-D-II, no.7, pp.1793–1801, July 1997.

[10] D. Marr and E. Hildreth, “Theory of edge detection,” Proc. Royal Soc., vol.B207, pp.187–217, London, England, Sept. 1980.

[11] N. Otsu, “A threshold selection method from gray-

level histograms," IEEE Trans. Syst. Man Cybern., vol.9, no.1, pp.62–66, Jan. 1979.

[12] A. Rosenfeld and A.C. Kak, Digital Picture Processing, 2nd ed., vol.2, Academic Press, San Diego, CA, 1982.

[13] F. Chang, C.J. Chen, and C.J. Lu, "A linear-time component-labeling algorithm using contour tracing technique," Comput. Vis. Image Understand., vol.93, no.2, pp.206–220, Feb. 2004.

[14] K. Suzuki, I. Horiba, and N. Sugie, "Linear-time connected-component labeling based on sequential local operations," Comput. Vis. Image Understand., vol.89, no.1, pp.1–23, Jan. 2003.

[15] M. Karnaugh, "The map method for synthesis of combinational logic circuits," Trans. AIEE., pt I, vo.72, no.9, pp.593–599, Sept. 1953.

（平成 19 年 7 月 17 日受付，10 月 9 日再受付）

何　　立風

1997 名古屋工業大学工学研究科博士後期課程了．博士（工学）．現在，愛知県立大学大学院情報科学研究科准教授．中国陝西科学技術大学客員教授．画像処理，定理証明，知識データベース，マルチエージェント等に興味をもつ．

巣　　宇燕

2000 名古屋大学大学院人間情報文化工学研究科博士後期課程了．2000 年 4 月から 2002 年 9 月まで名古屋工業大学研究員．現在，名古屋産業大学大学院環境マネジメント研究科准教授．博士（学術）．中国陝西科学技術大学客員教授．画像処理，定理証明及び図面理解，CAD などに興味をもつ．

鈴木　賢治

平 3 名城大・理工・電気電子卒．平 5 同大大学院修士課程了．同年（株）日立メディコ技術研究所入社．平 9 愛知県立大学情報科学部助手．平 13〜14 米国・シカゴ大放射線医科学研究科カートロスマン放射線像研究所客員研究員．平 14 同研究員．平 15 同研究講師，平 16 同研究助教授．平 18 同大放射線医科学研究科助教授．平 19 同大医用物理学研究科助教授兼任．平 19 同大がん研究センター助教授兼任．現在に至る．博士（工学）（名古屋大学）．コンピュータ支援診断，医用画像処理・認識及び機械学習の研究に従事．IEEE（Senior Member），AAPM，情報処理学会等各会員．平 14 Paul C. Hodges 賞，平 15 RSNA Certificate of Merit 賞，平 16 RSNA Research Trainee 賞，平 17 CRF Young Investigator 賞，平 18 SPIE Honorable Mention Poster 賞，RSNA Certificate of Merit 賞等受賞．

中村　剛士　（正員）

1998 名古屋工業大学工学研究科博士後期課程了．博士（工学）．現在，同大学情報工学科准教授．感性情報，ソフトコンピューティングなどに興味をもつ．人工知能学会会員．

伊藤　英則　（正員）

1974 名古屋大学大学院工学研究科博士課程了．工学博士号取得．同年 NTT 入社，横須賀研究所勤務．1985（財）新世代コンピュータ技術開発機構出向．1989 より名古屋工業大学教授．感性情報，自動推論，マルチメディア，マルチエージェントなどに興味をもつ．人工知能学会，ファジィ学会各会員．