コンピュータビジョン　52−3
（1988. 1. 21）

内容照合型ＬＵＴを用いたラベリング方式

LABBLING ALGORITHM USING CONTENT ADDRESSABLE LOOK UP TABLE

桝井　猛*　那須　靖弘*　志水　英二**
Takesi MASUI　Yasuhiro NASU　Eiji SIMIZU
*甲子園大学経営情報学部　Kosien University
**大阪市立大学工学部　Osaka City University

あらまし　筆者らはラベリング演算をパイプラインで処理するために、内容照合型ＬＵＴを用いて実現できるハードウエアの方式を考案した。従来、ラベリングの演算をラスタ走査型のハードウエアで実行する場合、ラベルの連結テーブルの統合処理での演算時間は、画像データに依存して一定でなく、ビデオレートで処理することは困難であった。そのため、ラベルの演算において、連結テーブルの内容を逐次更新することにより、連結テーブルの統合処理が不必要とする方式を考案した。本稿では、一次元テーブルによるアルゴリズムの説明し、つぎにテーブルの更新が可能な連結テーブルとして内容照合型ＬＵＴについて述べ、最後にパイプラインで処理するために必要なラベリングプロセッサを開発する上の問題点について報告する。

Abstract　We proposed a labeling algorizthm using Content Addressable look up Table (CAT) and operated by pipe-line. By using CAT , we can process a labeling operation in a constant rate. So in the labeling operation , the operate time dosen't depend on the variaties of the image data and we can use this algorithm for the video-rate operation.

In this paper, we explain a labeling algorithm using CAT and an outline of CAT which can modify labels by itself. We also discuss about problems to develop image processor system using CAT.

1．はじめに

近年、コンピュータの処理能力の拡大にともない、画像処理の分野も急速に実用化段階に近づき、ディジタル処理による多くの画像処理システムが発表されるようになってきた。これらの画像処理を対象としたシステムは、ソフトウェアでは計算量が膨大な処理に対して、ハードウエアからなる各種画像プロセッサを内部に持ち、高速処理を実現している。そのなかで最近、ＴＶカメラなどのビデオ機器からの映像信号を直接処理する、いわゆるビデオレートによる処理の実用化の研究も盛んになり、パイプラインを基本とした種々のシステムが開発されてきている。

筆者らは、これまでハードウエアによる高速処理が困難であったラベリングの演算に注目し、この演算を高速に処理する目的のために、２×２の近傍演算を基本とした局所並列型の演算回路とラベルの連結関係を格納できるテーブルを画像メモリ間に配置し、パイプラインで実現できるアルゴリズムとそのハードウエアを提案してきた。(1)～(3)

本内容は、連結関係のテーブルとしてラベルの更新が逐次行なえるＬＵＴを用いて、ビデオレート処理が実現できるラベリングのアルゴリズムに関するものである。二章では一次元テーブルを用いたアルゴリズムについて、三章では逐次ラベルの更新が行なえるＬＵＴの構成について、四章では具体的なハードウエアの構成について、五章ではＬＵＴの規模を決定するのに必要な初期ラベルについて、六章では総合評価

として実際にビデオレートが可能なハードウエアの問題点について、そのシミュレーション結果とともに報告する。

2．ラベリングアルゴリズム

本章では、いままで提案されてきた方式と比較しながら、筆者らが考案したアルゴリズムの内容を具体的に説明する。

2.1　ハードウエアによる処理方式

現在、画像処理を高速に実現する方法として、ハードウエアを用いた並列処理、パイプライン処理などが一般的に用いられている。従来、ハードウエアを用いたラベリングの演算方法として、第1図に示すように、

①局所並列のウインド演算で初期ラベルとラベルの連結関係を二次元テーブルに格納する（ハードウエア）。

②連結テーブルの内容をソフトウェアの処理などで統合する。

③最終ラベルを求める（ハードウエア）

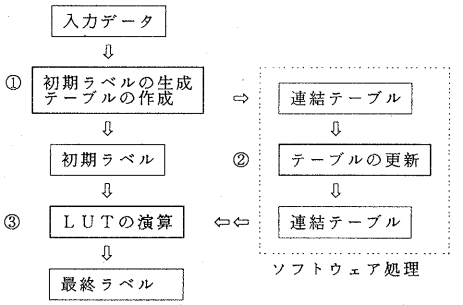
方法が提案されている。(7)　この中で①と③はハードウエア化による高速処理は容易であるが、②の処理（ラベルの統合）は演算時間がデータの内容に依存するため、一定時間内に演算を終了しなければならないビデオレートに対処することは容易ではない。そのため、これまでビデオレートでラベリングの演算を実現されたものは、②の処理時間が一定時間内におさまる入力画像データに対してのみ有効である。

筆者らは、入力画像のデータに依存しないでパイプラインが実現できる方式を開発するため、第1図に示した②の連結テーブルの統合処理を不要とする方法を検討した。第2図は筆者らが提案するラベリング処理のフローである。処理の流れは、

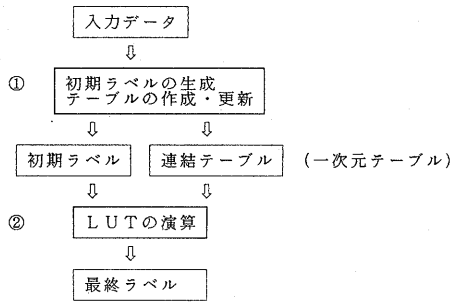
①初期ラベルの生成と連結テーブルの更新を同時に行なう

②連結テーブルをＬＵＴとして最終ラベルを生成する

の二段階の処理となる。②の処理は、従来の方法とまったく同じであるが、①で求めた連結テーブルがそのままＬＵＴとして使用できなければならない。このため、連結テーブルは一次元のデータ構造である必要がある。



第1図　ハードウエア処理



第2図　パイプライン処理

2.2　二段階のパイプライン方式

2.1で述べた二段階のパイプラインでラベリングを行う場合、次の2つの演算を同時に行なわなければならない。

・ラベルの生成・伝搬

・テーブルの作成

以下、二段階のパイプラインのための基本的な演算の内容を示す。

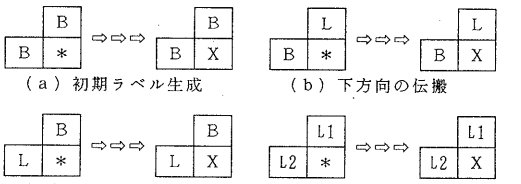
(1)　ラベルの生成・伝搬

入力画像に対して第3図に示す2×2のウインドを画像の左上から右下に水平方向に走査させ、1画素単位に演算を行う。（ａ）の演算により初期ラベルを生成する。そして、生成したラベルは（ｂ）～（ｄ）の演算によりそれぞれ右方向下方向に伝搬させる。また（ｄ）のように、上のラベルの値と左のラベルの値が異なるときは、どちらかのラベルを伝搬させる。

(2)　連結テーブルの作成

2×2のウインドで、（ｄ）のように上のラベルの値と左のラベルの値が異なるとき、二つのラベルの連結関係を格納する（たとえば、L1 と L2 のラベルが同じ領域のラベルである）ために、第4図に示すような一次元配列のラベルの連結テーブルが作成できる。テーブルのアドレスには、伝搬を中止するラベル値を対応させ、テーブルのデータには、継続して伝搬されるラベルの値を格納する。ここで格納されたラベルの値は、二段目のＬＵＴの演算で伝搬を中止された初期ラベルに置き換わる。これにより、この二つのラベルは一つの領域となる。ただし、(1)で初期ラベルが付くとき、ラベルの値の番地に同じ内容を格納する。

ここでラベル間の連結性は、新しい連結関係が成立するごとに拡大していくが、一段目の処理の終りで画像全体のラベルの連結関係が保持されていなければならない。ところで第



第3図　2×2の近傍演算

5図（a）のように(1)のラベルの伝搬においてラベル L1 の値とラベル L2 の値が異なるとき、テーブルのラベル L1 の値の番地にラベル L2 の値を書き込み、つぎに（b）のようにラベルの伝搬でラベル L1 の値とラベル L3 の値の連結関係が生じたとき、テーブルのラベル L1 の値の番地にラベル L3 の値を書き込むと、L1 と L2 の連結関係は消滅してしまう。このことにより、一次元の連結テーブルでは、二つのラベルの連結関係を単純に格納するだけでは、ラベル間の連結関係が維持できないことは明らかである。一次元のテーブルで連結関係を表現しようとする場合、これらの問題点を解決しなければならない。

2.3　一次元テーブル

二段階のパイプライン処理において、第2図の①で作成する連結テーブルを一次元のテーブルを用いて行うためには、

・連結関係が全て保持できていること。

・連結テーブルの内容が最終ラベルであること。

の二つの条件が完全に満たされなければ第2図の②でＬＵＴとして使用できない。ここでは、一次元テーブルについて検討する。

(1)　連結関係の保持

2.2の(2)で示した方式で連結関係をテーブルに格納する場合、二つのラベルの連結関係が保持できないパターンを検討した。第6図は2×2のウインドにおいて、上のラベルおよび左のラベルの値が異なった場合で、以下の三通りの伝搬の方法で連結テーブルを作成したものである。

（a）左のラベルの伝搬

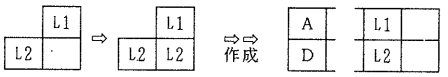
（b）上のラベルの伝搬

（c）優先ラベル（値の小さいラベル）の伝搬

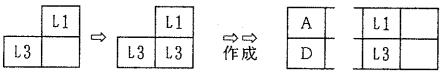
伝搬方法を3通り考察する理由は、ラベルの伝搬方法による影響があるかをしらべるためである。しかし、この例ではどのラベルを伝搬させても不完全な連結テーブルとなる。例えば、左のラベルの伝搬では、上から伝搬されるラベルが、上のラベルの伝搬では、左から伝搬されるラベルが、優先ラベルの伝搬では、値の大きいラベルの連結情報が格納できなくなることがわかる。

アドレス	伝搬を中止するラベルの値
データ	伝搬を継続するラベルの値

第4図　連結テーブル

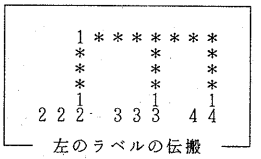


（a）L1 と L2 の連結



（b）L1 と L3 の連結

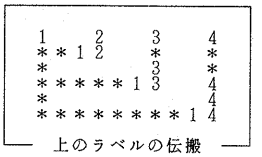
第5図　連結関係



A	1	2	3	4
D	X	2	3	4

X＝2、3、4

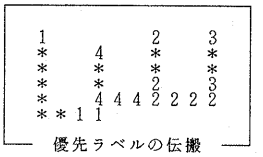
連結テーブル



A	1	2	3	4
D	X	2	3	4

X＝2、3、4

連結テーブル

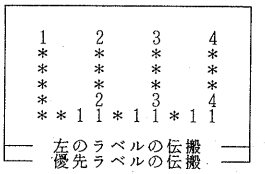


A	1	2	3	4
D	1	2	2	X

X＝1、2

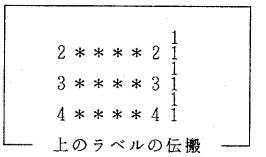
連結テーブル

第6図　連結関係が保持できない例



A	1	2	3	4
D	1	1	1	1

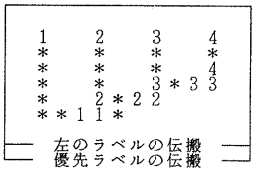
連結テーブル



A	1	2	3	4
D	1	1	1	1

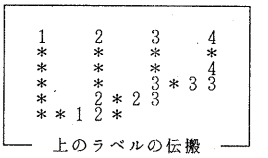
連結テーブル

（a）　最終ラベルが付く例



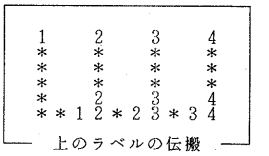
A	1	2	3	4
D	1	1	2	3

連結テーブル



A	1	2	3	4
D	2	3	4	4

連結テーブル



A	1	2	3	4
D	2	3	4	4

連結テーブル

（b）　最終ラベルが付かない例

第7図　最終ラベルの問題

⑵　最終ラベルの判断

連結テーブルに格納されているラベルの値は、⑴のラベルの生成・伝搬の終了時、最終ラベル（一つのセグメントは1つの値に対応しているラベル）でなければならない。第7図（ａ）は連結テーブルの値が最終ラベルである例で、第7図（ｂ）は連結テーブルの値が最終ラベルでない例である。ここでも、三つの伝搬方法どれをとっても、完全に最終ラベルにはならない。

2.2の方法において、連結テーブルの内容が最終ラベルにならない理由としては、テーブルに格納されているラベルの値は、新たに連結関係が生成しても更新されず、ラベルの連結関係の距離が拡大していくからである。たとえば、第8図に示すように、（ａ）でテーブルのラベル L1 の値の番地にラベル L2 の値を書き込む。つぎに（ｂ）のラベルの伝搬で、ラベル L2 の値とラベル L3 の値の連結関係が生じたとき、テーブルのラベル L2 の値の番地にラベル L3 の値を書き込む。そのとき、連結関係の距離を1にするためテーブルの更新処理として、ラベル L1 のテーブルの内容を L3 の値を格納する。このように、連結テーブルに格納されているラベルを、いつも更新していけば、一次元のテーブルで最終ラベルが保持できる。

⑶　テーブルの更新

第8図に示したように、連結テーブルを作成するうえで、新たに格納されるラベルの値が既に連結されている場合、連結しているラベルの番地の内容も更新する必要がある。このためには、連結テーブルすべての内容を検索しなければならないが、ビデオレートのパイプラインでラベリングの演算を行う場合、ラベルの更新に必要な時間は一般的にはとれないのが現状である。

新しいラベルを格納するたびに連結テーブルすべての内容を更新しない方法として、一行単位でテーブルを更新する方法が提案されているが、⑻　この方法では、連結テーブルの大きさが固定（２５６個）なので、ラベル数が多い画像データには対処することができない。

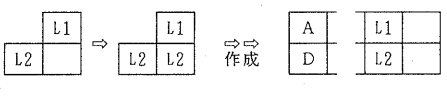
そこで、筆者らは、テーブルの大きさの拡張性があり、新たに格納されるラベルの値を、そのラベルに連結している全ての番地に格納できる機能をもつＬＵＴを考案した。以下、このＬＵＴを、「内容照合型ＬＵＴ」と呼ぶことにする。

⑷　テーブルの間接参照

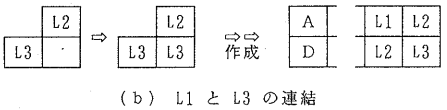
第9図は、ラベルの連結情報を内容照合型ＬＵＴに格納して、テーブルの更新を行なった場合の演算例である。

内容照合型ＬＵＴを連結テーブルに使用しても、（ａ）に示すように最終ラベルのテーブルが生成される場合と、（ｂ）のように最終ラベルが生成されない場合が生じる。（ｂ）の例はテーブルの更新がない場合、第7図（ｂ）に示すしたように連結情報は保持されるが、最終ラベルがテーブルに格納されていない画像データであり、（ｂ）の例はテーブルの更新がない場合、第6図に示すしたように連結情報が保持されない画像データである。このように、テーブルの更新だけでは、連結関係は保持できない。

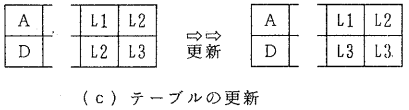
第9図のなかで、ラベルの連結関係が消滅される条件は第5図に示すように、伝搬を中止して既に伝搬ラベルが連結テーブルに格納されているラベルが、再度伝搬ラベルに、おきかえられるときにおこる。そのため、２×２のウインド演算



（ａ）　L1 と L2 の連結

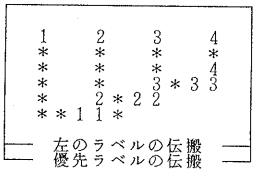


（ｂ）　L1 と L3 の連結



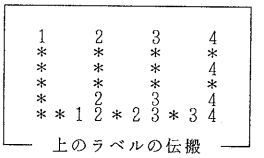
（ｃ）テーブルの更新

第8図　連結関係



A	1	2	3	4
D	1	1	1	1

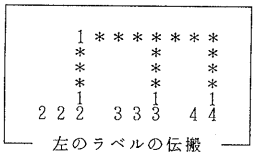
連結テーブル



A	1	2	3	4
D	4	4	4	4

連結テーブル

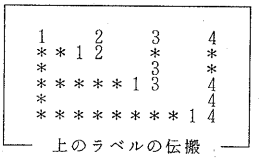
（ａ）　最終ラベルが付く例



A	1	2	3	4
D	X	2	3	4

X＝2、3、4

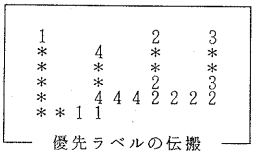
連結テーブル



A	1	2	3	4
D	X	2	3	4

X＝2、3、4

連結テーブル



A	1	2	3	4
D	1	2	2	X

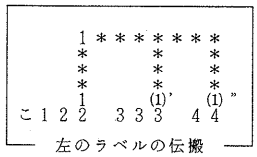
X＝1、2

連結テーブル

（ｂ）　最終ラベルが付かない例

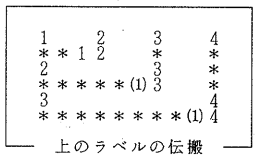
第9図　テーブルの更新

で比較されるラベルは、常に伝搬ラベルであれば連結関係は保持されるはずである。実際には、２×２のウインド演算を行うとき、ウインドに対応するラベルの値を連結テーブルをＬＵＴとして、その出力を用いれば常に伝搬ラベルとなる。第１０図はテーブルの更新しながら、２×２のウインド演算で入力データを連結テーブルに間接参照した例である。このように、一次元テーブルに「内容照合型ＬＵＴ」を使用し、２×２の演算において、入力とするラベルに「内容照合型ＬＵＴ」を間接参照させれば、2.3の二つの条件は満たされることになる。



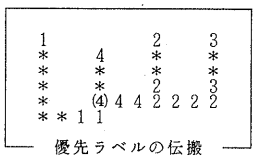
A	1	2	3	4
D	4	4	4	4

(1)' = 2、(1)" = 3
連結テーブル



A	1	2	3	4
D	4	4	4	4

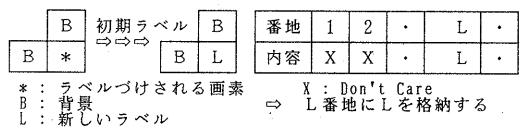
(1)' = 2、(1)" = 3
連結テーブル



A	1	2	3	4
D	1	1	1	1

(4) = 2
連結テーブル

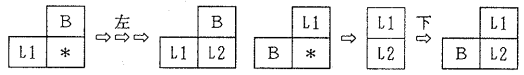
第１０図　テーブルの更新＋間接参照



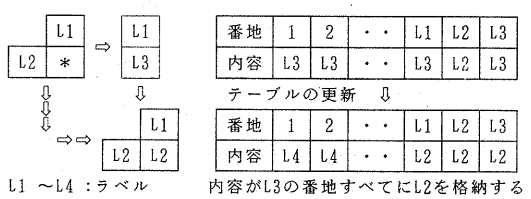
番地	1	2	・	L	・
内容	X	X	・	L	・

X：Don't Care
⇨　L番地にLを格納する

（ａ）初期ラベルＬがつく場合



（ｂ）ラベルの伝搬の場合



番地	1	2	・・	L1	L2	L3
内容	L3	L3	・・	L3	L2	L3

テーブルの更新　⇩

番地	1	2	・・	L1	L2	L3
内容	L4	L4	・・	L2	L2	L2

内容がL3の番地すべてにL2を格納する

（ｃ）初期ラベルL1とL2が異なる場合

第１１図　ラベリングの演算内容

2.4　演算内容

ここでは、2.3の方法に準拠して、「内容照合型ＬＵＴ」を用いた具体的な演算内容を示す。

(1)ラベルの生成

図１１(a) に示すように、従来と同様にしてラベルを生成する。このとき同時に、連結テーブルの初期化としてラベルの値に対応した番地にラベルの値を格納する。

(2)ラベルの伝搬

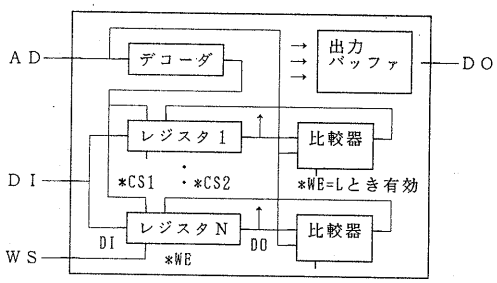
図１１(b) に示すように、(1)で作成した連結テーブルの内容を間接的に参照しながら従来と同様にして、右方向、下方向にラベルを伝搬させる。このときは、連結テーブルは変化しない。

(3)テーブルの更新をともなうラベルの伝搬

２×２のウインドにおいて、上のラベルの値と左のラベルの値が異なるとき、図１１(c) に示すように、連結テーブルの内容を参照しながら左のラベルを伝搬させる。連結テーブルに対しては、上のラベルの値をもつすべての番地の内容を左のラベルの値に同時に変更する演算を行う。

3．内容照合型ＬＵＴの構成

第二章で述べた演算方式では、内容を参照して、テーブルを更新するＬＵＴが必要となる。普通のアドレス参照型のメモリで構成したテーブルでは、同時に複数のデータを変更することは不可能である。そのため、第１２図に示すようにアドレスのデコードおよびレジスタの内容を参照して、同時に複数のレジスタに入力データを書き込む機能をもつＬＵＴを考案した。このＬＵＴは内部にＮ個のレジスタとＮ個の比較器をもち、各比較器の出力が、対応するレジスタの書込みストローブに接続している。データの書込みは、従来のアドレスに対応したレジスタおよび、アドレスの値とレジスタ内容が一致したレジスタに行なわれる。このため、レジスタに格納されている値が、入力アドレスの値と一致するすべてのレジスタに入力データを同時に書き込むことが可能となる。また、データの読み出しは、従来のメモリによるＬＵＴと同じで、アドレスの値がデコードされて、対応する１個のレジス



ＡＤ　：　アドレス入力
ＤＩ　：　データ入力
ＤＯ　：　データ出力
ＷＳ　：　ライトストローブ

第１２図　内容照合型ＬＵＴの構成

タの内容がデータ出力となる。このＬＵＴはラベリング演算における①初期ラベルの書込み、②ラベルの更新、③ＬＵＴの処理に対して、①および②は書込みモード、③は読み出しモードで使用する。

このＬＵＴの規模は、初期ラベルの数が２５６のとき、８ビットのレジスタが２５６個、８ビットの比較器が２５６必要である。初期ラベルの数に比例して大きくなる。いま、８ビットのレジスタが８０ゲート、８ビットの比較器が４７ゲートで構成できるとすると、

２５６×（８０＋４７）＝　３３０００

となり、約４万ゲートの規模のＬＳＩで構成できる。

最近の半導体の進歩からすると、実現不可能な規模ではないと思われる。しかし、初期ラベルの数が増加してくると第１３図に示すように回路規模はかなり増加する。内容照合型ＬＵＴを用いて、実際にラベリング・プロセッサを設計する場合、テーブルの規模が問題となる。対象とする画像データの複雑度に比例して初期ラベルの数が多くなるため、最適な規模のテーブルを決定することは難しい。

4. ハードウエアの構成

内容照合型のＬＵＴを用いたハードウエアは、第１４図に示すように従来の２×２のウインド演算回路にＬＵＴを追加して構成できる。

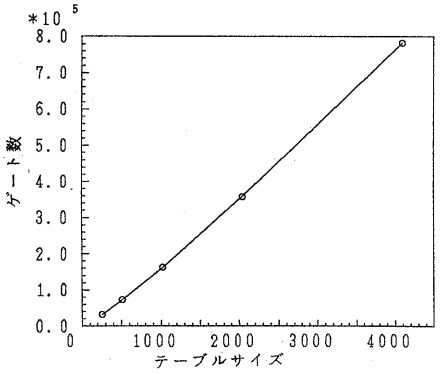
ウインドに対応する画素をレジスタR4、R3、および R2 に格納し、各々のレジスタ出力は判定回路に接続され、そこで２×２のウインドによるラベリング演算が行なわれ、伝搬ラベルの判定をする。各レジスタの値はマルチプレクサ MPX1 で選択され、伝搬ラベルとして外部に出力するとともに、レジスタ R3 に１つ前の画素の値として格納される。新しいラベルの生成される場合は、ラベルカウンタの出力を選択し、ラベルカウンタの値を１つ加算する。 R2 の出力はＬＵＴのアドレス入力 AD に接続し、そのデータ出力 DO を経由することにより間接参照され、また同時にレジスタ RO に一次格納される。そこで、上の画素（ R2 ）と左の画素（ R3 ）の値が異なった場合、ＬＵＴをライトモードにして、レジスタROに格納さられた値の番地に、レジスタ R3 の内容を書き込み、連結テーブルの更新が行なう。また、初期しいラベルが生成した場合は、マルチプレクサ MPX2 でＬＵＴのデータ入力を選択して、レジスタ R3 の値の番地に R3 の内容を書き込む。この回路は、ＬＵＴの演算のための入出力もある。

本回路は、２×２のウインド演算でＬＵＴのリードおよびライトの処理があり、画像入力の二倍のクロックが必要になるが、すべての処理は２サイクルで終了する。ビデオレートで画像データが入力されてきても、アクセスタイムの速いレジスタを持つならば、１画素の処理時間内で、ＬＵＴのリード・ライトは可能である。

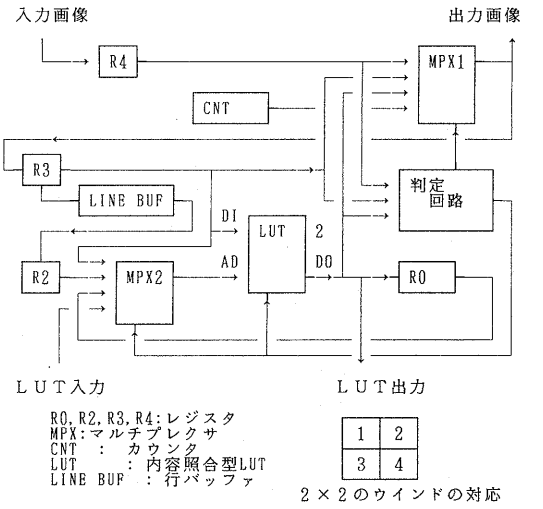
第１５図に内容参照型ＬＵＴを用いたラベリング・プロセッサの構成を示す。第１４図で示した２×２のウインド演算回路と１画面分の画像バッファを各々２組持ち、フレーム単位に回路をきりかえれば、ビデオレートで連続に入力してくるビデオ信号に対してもパイプライン処理が可能となる。

5. 初期ラベリングアルゴリズム

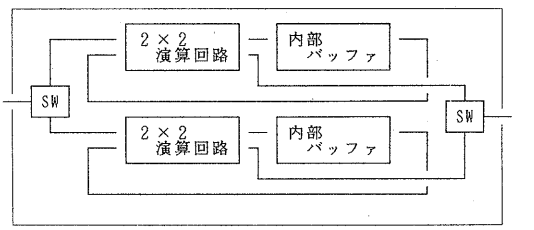
三章で述べたように、ラベリング演算のハードウエアの規模は、一段目の初期ラベル数に比例し、ＬＵＴの規模が小さ



第１３図　内容照合型ＬＵＴの規模



第１４図　２×２の演算回路



第１５図　ラベリング・プロセッサ

いほどハードウエアが小さくなる。ラベリング演算の機能を持つ画像処理システムを構築する場合、ＬＵＴの規模を小さくするため、初期ラベルの数を抑える工夫が必要となる。そのため、本章では、初期ラベルの数とラベリングのアルゴリズムの関係を述べるとともに、初期ラベルを抑えるアルゴリズムについて報告する。

5.1　ラベルの伝搬方向

第３図で示した２×２のウインドを走査させて演算をする方法では、画像全体の領域に対してラベルは下、右方向にしか伝搬しない。本来同じラベルが付く領域に対しても、異なったラベルが生成されるため、初期ラベルの数は最終ラベルより多くなる。第１６図は乱数で発生させた５００個のテストデータに対して、最終ラベルと初期ラベルの比を示したものである。画像データは１０通りの乱数を、２５６×２５６の大きさに発生させ、各々５０個の閾値で二値化したものである。横軸は、５０個の閾値で、縦軸は比である。最終ラベルの演算は、スパイダーのサブルーチンを用いて求めた。初期ラベル数の演算は、２×２のウインド演算のシミュレーションで求めた。テストデータによれば、約７００倍近い初期ラベルが発生することがわかる。

5.2　ラベルの伝搬方向の拡大

ラベルの初期ラベルの数を抑える方法においては、種々の方法が提案されているが、筆者らは２×２のウインド演算の拡張を考慮して、ラベルの伝搬の演算と同時に行なえるラベルの伝搬方向を拡大する方法を考案し、以下の２通りの伝搬方法の拡大方法についてシミュレーションを行なった。

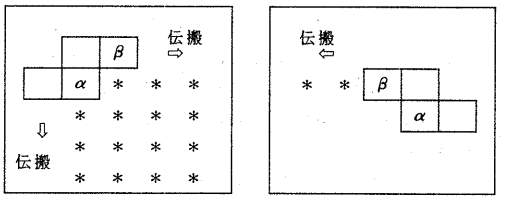
(1)左方向の拡大

先ずラベルの伝搬方向を第１７図に示すように左方向にも拡張したアルゴリズムを考案した。(4)　この方法では、ウインドによる近傍演算で、つぎの２つの処理を行なっている。

①　行きの処理として、画素αを対象とした演算を行ない通常のラベル付けを１行走査する。

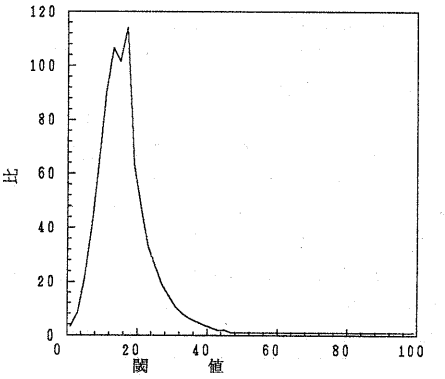
②　帰りの処理として、画素βを対象とした演算を行ないその行のラベルを左方向に伝搬する。

この二つの処理は並列に実行でき、またパイプライン処理が可能である。この場合の初期ラベルの最終ラベルの比を第１８図に示す。この場合では、最大値が１３０倍まで減少し

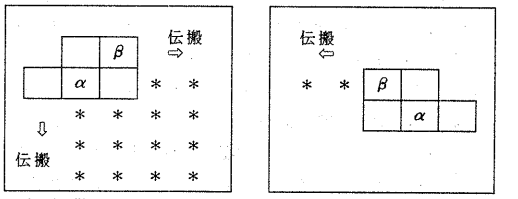


（ａ）従来の伝搬　　（ｂ）伝搬の拡張

第１７図　ラベルの伝搬方向の拡大（左）

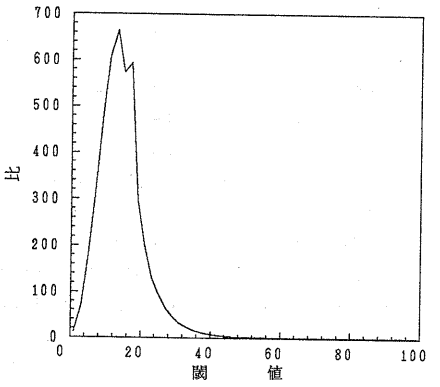


第１８図　初期ラベルと最終ラベルの比（左）

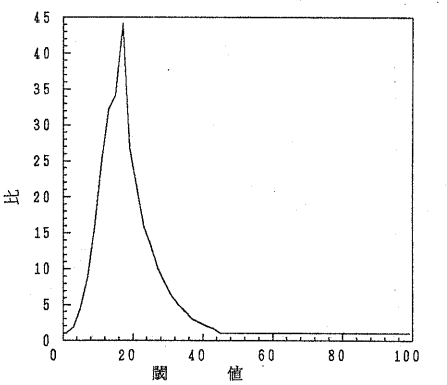


（ａ）従来の伝搬　　（ｂ）伝搬の拡張

第１９図　ラベルの伝搬方向の拡大（上）



第１６図　初期ラベルと最終ラベルの比



第２０図　初期ラベルと最終ラベルの比（上）

た。第１６図と比較すると、演算回路は複雑になるが、最大で５分の１に減少している。

⑵上方向の拡大

つぎにラベルの伝搬方向を第１９図に示すように上方向にも拡張したアルゴリズムを考案した。⑹ この方法も、ウインドによる近傍演算で、つぎの２つの処理を行なっている。

① 通常の走査において、画素αを対象としてラベルの伝搬だけを行なう。

② 帰りの処理において、画素βを対象としてラベルの生成と伝搬を行なう。

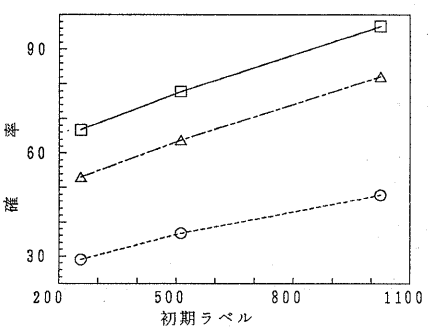
この二つの処理もパイプライン処理が可能である。この場合の初期ラベルの最終ラベルの比を第２０図に示す。最大値が５０倍になり、ラベルの伝搬の拡張がない演算に比較すると、初期ラベルの数が約１０分の１に減少している。

5.3 初期ラベルの減少

5.2のウインド演算によるラベルの伝搬方向の拡大で、連結テーブルに格納される初期ラベルの数を減らすことが可能であることが判明した。初期ラベルの数が減少すれば、ＬＵＴの規模も小さくなり、ハードウエアの負担も減少する。その他、初期ラベルを減らす方法として、種々の方法があるので、ＬＵＴと組合せて評価する必要がある。しかし、ウインド演算と同時に行なえる方法でなければ、逆にハードウエアが複雑になるので、さらに総合的な方法を考案する必要がある。

6. 総合評価

実際にラベリング・プロセッサのハードウエアを用いて、画像処理システムを構築する場合、ハードウエアの規模が問題となる。第五章で示したテストデータすべてをカバーするには、初期ラベルを減少させる工夫が更に必要になる。しかし、画像処理の実用化の面から考えると、対象とする画像データに依存したハードウエアを構成することも検討しなければならない。そこで、サンプル数は少ないが第五章で使用した５００個のテストデータの中で、最終ラベルが２５６以下になるデータを用いて、初期ラベルの分布の確率を求めた。第２１図は２０７個（全体の４１％）のデータに対する初期ラベルの分布の確率である。○は、初期ラベルの抑制をしない場合、△は初期ラベルを左に伝搬させた場合、□は初期ラベルを上方向まで伝搬させた場合である。



第２１図 ２５６以下の画像の初期ラベルの確率

初期ラベルの抑制をしない場合は、２５６ワードのテーブルでは全体の３０％、１０２４ワードのテーブルを使用しても全体の５０％のデータしか演算結果が保証されない。初期ラベルの伝搬で、上方向の伝搬までの演算を行なえば、２５６ワードのテーブルでは全体の６６％、１０２４ワードの初期ラベルテーブルで全体の９５％の画像に対して、演算結果が有効になることがわかる。これより、ラベル数が２５６以下の画像に対して、初期ラベルの抑制を効果的に行なえば、実際に必要なＬＵＴの規模は、最大１０２４ワードで十分実用になると思える。

7. あとがき

２段階のパイプライン処理で実現できるビデオレート可能なラベリング方式を考案した。本方式では、初期ラベリングの演算において、最終ラベルの連結情報を格納したテーブルが求まり、初期ラベリングの演算終了後、そのテーブルをＬＵＴとして演算でラベリング出力が得ることができる。この方式は、二段階のパイプライン処理であり、ハードウエアを二組用いれば、ビデオレートの画像に対しても連続に演算が可能となる。ここで用いた内容照合型ＬＵＴは、ラベル数が８ビットの場合、レジスタ、比較器などの部品が各々２５６個で構成でき、約４万ゲートの規模であるので、現在の半導体技術を用いれば、ＬＳＩ化は可能であると思える。

謝辞 日頃、研究に御世話になる大阪市立大学工学部電子回路研究室の諸氏に深謝します。

参考文献

（１）桝井他、”ハードウエアを用いたラベリングの高速化手法Ⅰ”、昭和61年電気関係関西連合大会

（２）桝井他、”ＬＵＴを用いたラベリングの演算方式”昭和62年電子情報通信総合全国大会1206

（３）桝井他、”内容照合型ＬＵＴを用いたラベリングの演算方式”昭和62年電子情報通信部門全国大会 249

（４）那須他、”ハードウエアを用いたラベリングの高速化手法Ⅱ”、昭和61年電気関係関西連合大会

（５）那須他、”ハードウエアのための初期ラベリングアルゴリズム”昭和62年電子情報通信総合全国大会1207

（６）那須他、”ハードウエアのための初期ラベリングアルゴリズム（Ⅱ）”昭和62年電子情報通信部門全国大会

（７）後藤他、”パイプライン型高速ラベル付けアルゴリズム”情処学会ＣＶ研究会資料ＣＶ４３－１（1986）

（８）服部他、”ビデオ・レートで連続画像処理する高速ラベリング・プロセッサ”昭和62年後期情処学会全大会