社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2004－CVIM－144 (4)
2004／5／6

４分木表現を用いた画像の高速最近傍識別

武本 浩二　加藤 丈和　和田 俊和

和歌山大学システム工学研究科 〒640-8510 和歌山市栄谷 930

E-mail: {takemoto,tkato,twada}@vrl.sys.wakayama-u.ac.jp

あらまし 本論文では，画像から色ターゲット検出などによって得られた画像領域に対して，最近傍識別を高速に行う方法について述べる．本手法では，画像領域を表す２値画像に対して，あらかじめ登録しておいた大量のプロトタイプと入力とを比較することで，最近傍識別を行う．まず，検出された２値画像領域を４分木で表現し，対象領域と非対象領域を両方含むグレーノードに密度情報を記録しておくことにより，任意の解像度で入力画像と各プロトタイプ間の距離の上限・下限値を求め，解像度を順次高めながら分枝限定法やA*アルゴリズムと同様の基準で効率の良い枝刈り計算を行う手法を提案する．次にあらかじめプロトタイプをグループ化することにより，初期の段階でも全プロトタイプと入力との距離計算を回避し，プロトタイプ数が増加しても速度低下が起きにくい２値画像に対する最近傍識別を提案する．

An Accelerated Nearest Neighbor Classification for Binary Images using Quadtree representation

Koji TAKEMOTO　Takekazu KATO　and　Toshikazu WADA

Graduate school of System Engineering, Wakayama University

930 Sakaedani, Wakayama-city, Wakayama, 640-8510 Japan

E-mail: {takemoto,tkato,twada}@vrl.sys.wakayama-u.ac.jp

Abstract This paper presents an accelerated Nearest Neighbor(NN) classifier for binary images generated by color target detection. This method finds the NN prototype to an input image from many prototypes and classifies the input to the prototype class. For this purpose, we employ an extended quad tree representation of a binary image, in which the density values are embedded into the grey nodes. In the coarse-to-fine comparison between two trees, we can calculate the upper and lower bounds of the distance between them. By using these boundaries, we can reduce the NN candidates to the input similarly with the branch-and-bound and/or A* algorithms. At a coarse level, the number of NN candidates is big, but the computational cost of each comparison is low. As well, each comparison at a fine level is computationally expensive, but the number of candidates is small. This balancing mechanism accelerates the NN classification. Next, we propose a further acceleration method of NN classification by clustering the prototypes, which suppresses the speed down caused by prototype population increasing.

1. はじめに

最近傍識別[1]は 1967 年に提案された最も古典的な識別方法であるが,次に示す優れた特長があるため，現在でもその研究が行われている．

1. トレーニングデータが十分与えられている場合，ベイズ誤り確率の二倍以下の誤り確率を達成できる．
2. 他の識別器のように確率分布モデルなどの事前知識を必要としない．これに対し，例えば Support Vector Machine (SVM)[3] では kernel 関数，ADA Boosting[4]では base classifier などの事前知識を必要とする．
3. 最大マージン基準で識別を行った場合と同等の識別結果が得られる．

4. そのままで多クラスの識別が行える.

最近傍識別ではパターン分布をモデル化するといったことは行わず, 事例のみで識別を行うため, Case (or Instance) Based Reasoning (CBR or IBR)[2]の中核技術として位置づけられている. しかし最近傍識別では, 大量のトレーニングデータを与える場合や, データが高次元である場合に, 計算量が増し実行に時間がかかりすぎる. このような問題に対し我々は, 識別性能を落とさずに記憶するパターン (プロトタイプ) の数を削減する Condensing の高能率化[5], これを利用した識別器の高速化[6]などの研究を行ってきた.

本論文では、最近傍識別を２次元の画像領域に対して適用する方法について議論する. 一般に画像を認識する際に, 画像をラスタースキャンによってベクトルデータに変換して扱うことが多いが, これは, 画像観測時に拡大・縮小, 平行移動などの変換が起こるため, 比較する画像に応じて画像の表現を変更する必要があるため得策ではない. 例えば, サイズの異なる画像を比較する際には大きさを揃えて比較する必要がある. 平行移動に関しても同様である. このように変換パラメータを動的に変更することにより, ベクトルの次元数や, 要素の順序が変化するため, 画像をそのままベクトル表現して最近傍識別を行うことは望ましくない.

画像に含まれる部分パターンを認識する問題は, 変換パラメータの探索を行いつつ, 識別を行う問題として定式化される. しかし, 拡大や回転までを含めて探索を行うのは, 探索空間が広くなり効率的ではない. 画像の拡大・縮小に関しては, 異なる大きさの画像を比較するため, 画像の 4 分木表現が有効であることが知られている. 4 分木表現を用いることにより, 画像間の類似性・相違性を階層的に評価することが可能になり, 画像サイズの食い違いは問題にならなくなる.

このように, 画像の拡大・縮小が起きる下での最近傍識別を行うには, 4 分木表現が有効であるが, 従来は 4 分木間の構造的類似性・相違性の尺度を定義して識別を行うという手法が研究されてきた. これに対して我々は, 2 値画像間に定義された距離に基づく最近傍識別のために 4 分木表現を用いる方法を検討してきた[7]. これは, 画像間距離を, 画素値が食い違う画素の総数すなわち, 画像間のハミング距離とした場合, 4 分木上でこの距離を階層的に評価し最近傍候補を枝刈りによって絞り込みながら識別を効率的に行う方法である.

この階層的な枝刈りにより, 図 1 に示すように, 低い解像度では多数のプロトタイプとの比較を行うが個々の比較の計算コストは低く, 高い解像度では計算コストは増加するが比較する候補数は絞り込まれている, というように効率の良い探索を行えるようになる.

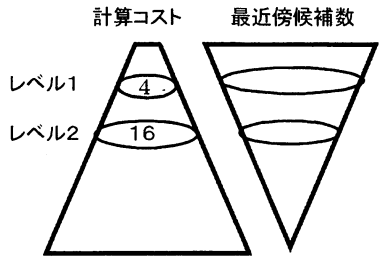

図 1 . 解像度と候補数の関係

さらに, Best-First Search を行うことによって暫定値を急速に下げ, より効率的な枝刈りを実現する方法, および最近傍パターンの探索ではなく最近傍識別であるという問題の特殊性を利用して高速化を実現する方法を提案した.

この手法は2値画像の高速な最近傍識別を行う非常に有効な方法ではあるが, 記憶するプロトタイプ数が増加すると, 初期レベルでの比較演算が増加するため, 大量のプロトタイプが与えられる問題に対して識別時間が低下するという問題点がある. このような問題を解決する方法として, データ間の距離情報を最大限に利用してデータ集合の枝刈りを試みる方法[9]やプロトタイプを階層的にクラスタリングする方法[10] などがあるが, 本論文では後者のアプローチを応用した手法を提案する. これはあらかじめプロトタイプの階層的クラスタリングを行い, 入力データとクラスタ間の距離を階層的に評価しながら最近傍プロトタイプを含み得ないクラスタの枝刈りを行うことによって, 全プロトタイプと入力の画像間の距離評価を行うことを回避する手法である.

2. 使用する記号とその定義

以下の議論では次の記号を用いる.

画像領域：

$$D^n = \{Z \cap [1,2^n]\} \times \{Z \cap [1,2^n]\}$$

但し Z は整数の集合, $\times$ は直積.

部分画像領域：

$$D^m_{l,u} = \{Z \cap [l,l+2^m-1]\} \times \{Z \cap [u,u+2^m-1]\}$$

2 値画像： $I(x,y) \in \{0,1\}$, $x,y \in D^n$

部分 2 値画像： $I^m_{l,u}(x,y) \in \{0,1\}$, $x,y \in D^m_{l,u}$

画素数： $|I^m_{l,u}|_B$: $D^m_{l,u}$ 中で値が B の画素数

画素値の ExclusiveOR 演算： $I_1 \oplus I_2$

画像間距離： $d(I_1, I_2) = |I_1(x,y) \oplus I_2(x,y)|_1$
これは値が異なる画素の総数を表している．

２値画像間の類似性の評価は，画像間の差分を用いる．２値画像間の差分は，画像間の排他的論理和 $I_1 \oplus I_2$ を計算し，画素値が異なる画素の総数 $|I_1 \oplus I_2|_1$ を求めることにより計算する．この２値画像間の差分を画像間距離 $d(I_1, I_2)$ とする．

3. 画像の４分木表現による効率的最近傍識別

画像の４分木表現は，$D^n_{l,u} = D^n_{1,1}$ からスタートして，画像領域 $D^m_{l,u}$ 内の画素が黒画素もしくは白画素のみになるまで $D^m_{l,u}$ を $D^{m-1}_{l,u}$，$D^{m-1}_{l+2^{m-1},u}$，$D^{m-1}_{l,u+2^{m-1}}$，$D^{m-1}_{l+2^{m-1},u+2^{m-1}}$ に四分割することを再帰的に繰り返して得られる．

４分木の根をレベル０，１段深くなるごとにレベル 1,2…とする．この場合，レベル m のノードは１辺の長さが 2^{n-m} の部分画像領域に対応している．この画像領域内の黒画素と白画素の両方を含む画像領域は，Gray ノードと呼ばれ４分木の分枝節に相当する．黒画素あるいは白画素のみから成る画像領域は黒／白ノードと呼ばれ，木構造の葉節に対応している．

４分木表現は最初の分割を行う位置（分割開始点）によって構造的に大きな変化が生じるという問題点がある．本研究では，「入力画像中の対象領域が検出されている」ことを前提とする．この前提の下では，対象領域の重心位置が計算可能であり，重心を分割開始点として用いることにより，上述の問題を回避することができる．具体的には以下のように４分木を計算する（図２参照）．

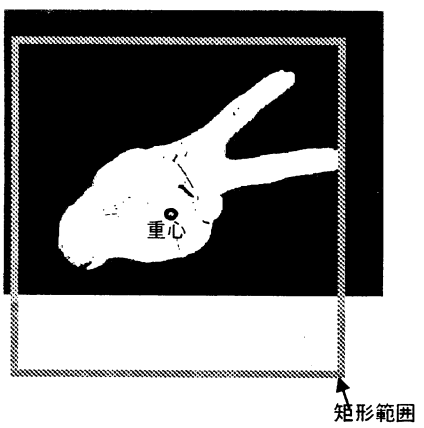

図２．矩形範囲

１．分割開始点は対象領域の重心にする．
２．分割する矩形範囲（Bound Box）は領域の重心を中心として，対象が全て収まる正方領域とする．

また，対象の回転に対しても同じように，長軸方向をそろえるなどの正規化を行うことによって回転した対象を同じクラスと見なして登録する．回転した対象を同じ対象と見なさない場合は，回転の正規化を行なわない．

3.1. 4 分木の比較による画像間距離の評価

画像間距離 $d(I_1,I_2)$ は，直接画像同士の比較を行わなければ計算することができない．しかし，プロトタイプ集合 P の中から入力 I_{in} に対する最近傍プロトタイプ $I^* = \arg\min_{I \in P} d(I_{in}, I)$ を探索する問題で，直接画像の比較を行なうことは極めて効率が悪い．そこで，４分木を用いて粗い解像度から順次解像度を上げながら $d(I_{in},I)$ の上限値と下限値を推定し，I^* の候補と成り得る画像集合を絞り込むという方法を採用する．この方法は，低い解像度では多数のプロトタイプとの比較を行うが個々のデータは少なく，高い解像度では個々のデータが増加するが候補数は少数に絞り込まれている，という効果が得られるため効率の良い探索が行える．

$d(I_{in},I)$ の上限と下限を計算する必要があるため，部分画像 $I^{n-m}_{l,u}$ に対応するレベル m のノードに，次式によって計算される（白画素の）密度情報 $\delta^m_{l,u}$ と $D^m_{l,u}$ の面積に相当する重み W_m を記録する．

$$\delta^m_{l,u} = \frac{|I^{n-m}_{l,u}|_1}{|I^{n-m}_{l,u}|_0 + |I^{n-m}_{l,u}|_1}, \quad W_m = 4^{n-m}$$

3.1.1. 木構造同士の比較による上限下限の計算

このようにして得られた２つの４分木の同一レベルノードを比較することにより，画像間の距離を推定する問題を考える．まず，レベル m の対応するノード間の比較結果を，その総和が $d(I_1, I_2)$ になるように定義する．密度 δ_1, δ_2 を持つノード間の比較結果 $c_{1,2}$ は次のように計算できる．

- 黒もしくは白ノード同士： $c_{1,2} = 0$
  この場合，総ての画素が一致するため，この部分では距離０となり，食い違う画素数も０となる．
- 黒と白ノード： $c_{1,2} = W_m$

この場合，総ての画素が食い違うため，レベルmでは，W_m画素の食い違いとなる．

- 黒とGrayノード：　$c_{1,2} = \delta W_m$

  黒画素のうち一致する割合は$1-\delta$，一致しない割合は，δである．これにW_mをかけたものが食い違う画素数となる．

- 白とGrayノード：　$c_{1,2} = (1-\delta)W_m$

  白画素のうち一致する割合はδ，一致しない割合は，$1-\delta$である．これにW_mをかけたものが食い違う画素数となる．

- Grayノード同士：　$c_{1,2(\min)} = |\delta_1 - \delta_2| W_m$

  $c_{1,2(\max)} = \max\{\delta_1+\delta_2, 2-(\delta_1+\delta_2)\}W_m$

最も画素値が一致すると場合，$|\delta_1-\delta_2|W_m$が食い違う画素数となり，図3のように一方の領域にもう一方の領域が完全に含まれ▨の領域が食い違う．最も食い違う場合は，$\delta_1+\delta_2 \le 1$のとき$(\delta_1+\delta_2)W_m$，$\delta_1+\delta_2 \ge 1$のとき$\{2-(\delta_1+\delta_2)\}W_m$が食い違う画素数となり，図4のように完全に領域は重ならず▨の領域が食い違う．

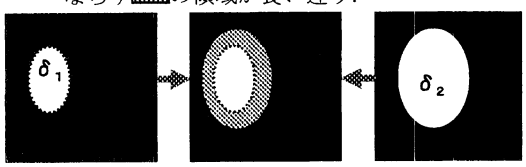

図3．最も画素が一致した場合

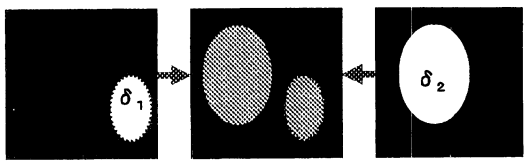

図4．最も画素が食い違う場合

このようにして求めた$c_{1,2}$を同一レベルのノードについて足し合わせることにより$d(I_1,I_2)$が求められるが，Grayノード同士の比較結果は，最大値$c_{1,2(\max)}$と最小値$c_{1,2(\min)}$しか求めることができないため，$d(I_1,I_2)$も上限値，下限値しか決まらない．$d(I_1,I_2)$の上限値$\overline{d}(I_1,I_2)$の計算には，Grayノード同士の比較結果の最大値$c_{1,2(\max)}$を用い，下限値$\underline{d}(I_1,I_2)$の計算には，最小値$c_{1,2(\min)}$を用いノード間距離の総和によって求められる．このとき，

$\underline{d}(I_1,I_2) \le d(I_1,I_2) \le \overline{d}(I_1,I_2)$が成り立つ．また，距離の不確定幅$a(I_1,I_2) = \overline{d}(I_1,I_2) - \underline{d}(I_1,I_2)$は，レベルに対して単調減少し，レベル$n$ではGrayノードが無くなるため0に収束する．

3.1.2. 暫定値を用いた枝刈り

入力I_{in}に対して求めた上限値の最小値

$$g = \min_{I \in P} \overline{d}(I_{in}, I)$$

を暫定値とおく．これは，「最悪の場合でも入力とプロトタイプ集合Pの要素の最短距離はg以下になる」ということを表している．したがって，$g < \underline{d}(I_{in}, I)$となるプロトタイプ$I$は$I_{in}$の最近傍とはならないことが保障され，これらは最近傍プロトタイプの候補から除外できる．例えば図5に示した例では，I_BとI_Dが最近傍候補から除外される．

レベルを上げて計算した暫定値は，距離計算の不確定幅がレベルに対して単調減少することから，増加することはない．つまり，より小さな暫定値が得られるたびにgを更新することができる．この性質を利用して候補数を削減しながらレベル0からレベルnまでの計算を行えば，I^*を効率よく求めることができる．

3.1.3. Best-First Search を用いた効率化

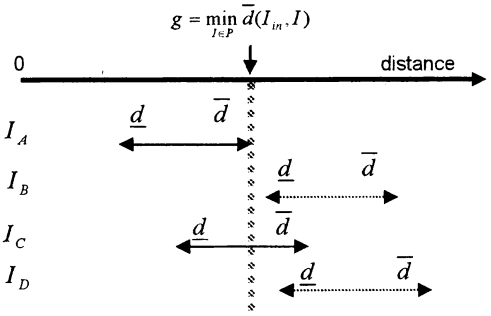

図5．暫定値による枝刈り

暫定値を用いることにより，効率的な探索が実現できるが，レベルを上げた場合に暫定値の減少速度が遅いケースが考えられる．このような場合には，入力I_{in}と，最も近くなる可能性があるプロトタイプ，すなわち$\underline{d}(I_{in}, I)$が最小になるプロトタイプIとの間でレベルを上げた比較計算を先に行い，暫定値gを積極的に下げるという方法が有効であると考えられる．このようにして得られた暫定値を用いることにより，レベルの低い段階でより効率的に最近傍候補を絞込むことができる．

3.1.4. 最近傍識別問題の性質を利用した効率化

本研究では最近傍識別問題を扱っており，必ずしも最近傍プロトタイプを探す必要はない．した

がって，最近傍候補が全て同一クラスに属するプロトタイプとなった時点で，識別結果を出せばよい．このような計算の変更により，木ノードの比較計算回数が減少し，より高速な識別計算が行えるものと考える．

4. プロトタイプのグループ化

さらなる効率化のために，実行の初期段階で，全てのプロトタイプとの距離計算を行う問題を回避するために，あらかじめプロトタイプをクラスタリングによってグループ化し，入力に対する最近傍プロトタイプを含み得るグループの絞り込みを行うことによって入力との比較回数を減らす方法について提案する．

4.1. クラスタリングによるグループ化

プロトタイプの絞り込みを用いて画像の最近傍識別を行っても，プロトタイプが増加すると実行時間も増加してしまう．実行の初期の段階では解像度が低い状態の計算ではあるが，全プロトタイプとの比較を行っているので，最終的な実行時間はプロトタイプ数に比例したものとなる．そこで，プロトタイプをグループ化し，グループの絞り込みを行うことにより，比較の回数を減らす方法を提案する．

このグループの絞り込みの実行時間はグループ数に比例したものになると考えられる．グループ数はプロトタイプ数が少ない場合プロトタイプ数に比例するが，プロトタイプ数がある程度増加するとその増加は抑えられ，実行時間も抑制されるはずである．

本論文では，画像間の相違度を，画像間距離と見なし，最近傍画像を相違度最小の画像として扱っているため，グループ化を行う際の判断基準も画像間距離を用いるべきである．そこで，本論分では，グループ化の方法として，クラスには関係なく画像間距離に基づいたクラスタリングを行う．クラスタリングの方法としては，K-Means クラスタリングを用いる．以下の議論では次の記号を用いる．

- クラスタ：C_i
- クラスタ半径：$r(C_i)$
- 近傍クラスタ：C_n
- 近傍プロトタイプ：p_n
- クラスタ暫定値：$g_c = d(C_n, I_{in}) - r(C_n)$

あるクラスタ C_i に属するプロトタイプ数が多いほど C_i が候補から外れた場合，最近傍候補から除外されるプロトタイプ数は多くなる．しかし，クラスタに属するプロトタイプが増えるとそのクラスタ半径 $r(C_i)$ は大きくなり，距離計算を行う際の曖昧性が大きくなるので，絞り込みが難しくなる．そこで，クラスタ数は２つにし，２分割を再帰的に繰り返すことによって，クラスタの２分木を作成することにより，クラスタの絞り込みが階層的に行えるようになる．

以下にその具体的な方法を述べる．ただしクラスタ中心はそれに属するプロトタイプの重心とする．

1. クラスタ中心を２つランダムに決める．
2. 各プロトタイプについてクラスタ中心からの距離を評価し，距離が近いクラスタをそのプロトタイプの帰属クラスタとする．
3. クラスタ毎にプロトタイプの平均値を求め，それをクラスタ中心として置き換える．
4. クラスタ中心が移動していれば，２に戻る．
5. ３の置き換えによって，２つのクラスタそれぞれについて再帰的に１からのクラスタリングを行う．プロトタイプが１つなら終了する．

これにより作成したクラスタの２分木を用いて階層的にクラスタの絞り込みを行う方法を考える．クラスタの中心はプロトタイプと同様に４分木表現を用いることにより，クラスタにおいても階層的な距離計算が可能となる．

4.2. クラスタの絞り込み

プロトタイプの絞り込みと同様，クラスタの絞り込みにおいても先に暫定値を決める．ある入力 I_{in} が与えられると，あらかじめ作成しておいたクラスタの２分木を用いて，I_{in} と近いクラスタを探索し，それを基にクラスタを絞り込むための暫定値 g_c を決定する．ここで，出来るだけ近いクラスタを探すことが望ましいが正確な距離計算を行い探索を行うことは時間の無駄であり，いい検索結果が得られるとも限らない．

そこで以下のように I_{in} と近い可能性の高いクラスタを探索した．

1. クラスタの２分木のルートから探索を始める．
2. ２つに分かれたクラスタ C_i，C_{i+1} と，I_{in} との距離を，４分木を活かしてレベルの低い状態で計算し，距離の上限・下限値を求める．
3. 求めた上限・下限値の中間値を I_{in} との仮の距離とし，この仮の距離が小さいほうのクラスタに進む．
4. ２・３を繰り返す．

この方法により，短時間である程度 I_{in} と近いと思われるクラスタ C_n を探索できる．

この C_n と I_{in} との距離 $d(C_n, I_{in})$ に C_n のクラスタ半径 $r(C_n)$ を足したものをクラスタ暫定値 g_c と呼ぶ．この g_c は「少なくとも，I_{in} とプロトタイプ集合 P の要素の最短距離は g_c 以下になる」ということを表している．この g_c はできるだけ小さい方が望ましいので，$r(C_n)$ が最も小さくなる，すなわちプロトタイプが１つになるまで探索を続け，そのクラスタが近傍クラスタとなる．この近傍クラスタ内の１つだけあるプロトタイプを近傍プロトタイプ p_n とすると，

$g_c = d(P_n, I_{in})$ となる．

この g_c を用いて各クラスタ C_i が候補から外れるかを判定する．I_{in} と C_i との距離は $d(C_i, I_{in})$ となり，そのクラスタ C_i 内のプロトタイプと I_{in} との距離は必ず $d(C_i, I_{in}) - r(C_i)$ 以上となる．このことを用いて，

$$g_c < d(C_i, I_{in}) - r(C_i) \quad (1)$$

となる C_i は最近傍のプロトタイプを持たないと言えるので，候補から除外できる．例えば図６ではクラスタ C_i は上の条件を満たしており候補から除外できる．

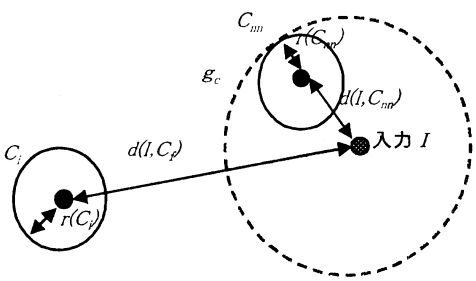

図６．クラスタの絞り込み

4. 3. クラスタの絞り込み計算の効率化

クラスタの絞り込みで，全てのクラスタについて絞り込みの計算をするのは効率が悪い．そこで（１）の式を変形すると，

$$d(C_i, I_{in}) > g_c + r(C_i)$$

となる．$r(C_i)$ は既知であり，もしこの $r(C_i)$ が大きい場合は，式が成り立つ可能性が少ないので，ある閾値を設け $r(C_i)$ がこの閾値より大きい C_i に関しては候補から外れる可能性が少ないとみなし $d(C_i, I_{in})$ を求めず，さらに下のクラスタをみることにより，無駄な距離計算が省ける．

4. 4. クラスタ暫定値の更新

クラスタ暫定値 g_c を決定する際の近傍プロトタイプ p_n の探索は正確なものではなく，g_c も必ずしも最適な値とは言えない．g_c は小さい程，絞り込みの効果は大きいといえるので，絞り込みの計算を行う際にもこの g_c の更新を試み，積極的に g_c を下げることにより計算の効率化を行う．

5. 実験

次の条件で実験を行った．

- トレーニングデータは，サイズが３２×３２の０～９の手書き画像（[11] の "Optical Recognition of Hardwritten digits" 使用）
- 入力画像は，トレーニングデータとは別に，０～９の手書き数字 10 枚（図７）を用意し，それぞれの入力の認識を行った結果の平均を実行結果とする．

図７．入力画像として実験に用いた手書き数字

5.1. 実行時間の比較

リアルタイムでの認識を行うためには，実行時間の短縮が重要である．ここでは，提案手法による速度を確かめるために実行時間の比較を行う．比較を行う手法は以下のうち Method 1～4 の４種類である．

- Method 1: 2 値画像での距離計算
- Method 2: 4 分木で距離計算（暫定値によるプロトタイプの絞り込み）

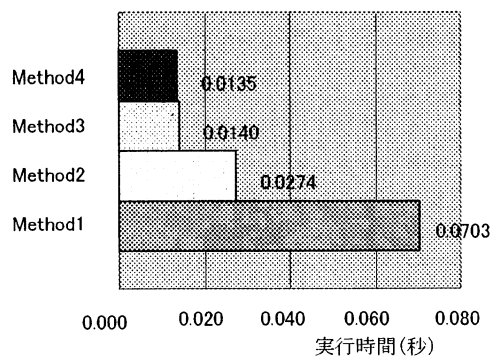

図８．各手法の実行時間

・ Method 3: Method 2＋Best-First Search によるプロトタイプの絞り込み
・ Method 4: Method 3＋識別結果確定時に終了

まず，5000 個のプロトタイプを登録し，実行時間を計測した結果を図８に示す．表１には，このときの Method 1 に対する各手法（Method 2～4）の高速化の割合（Method 1／Method 2～4）の値を示す．

Method 1 と Method 2～4 の実行結果を比べると，それぞれ高速化が実現できており，各手法の有効性が確認できる．

手法	高速化の割合
Method 2	2.58 倍
Method 3	5.81 倍
Method 4	5.99 倍

表 1. Method 1 に対する高速化の割合

5.2. プロトタイプ数に対する実行時間の変化

プロトタイプ数を 1000，2000，3000，4000，5000 と変化させた場合の Method 1 と Method 4 の実行時間は図９のようになった．

この図９から，Method 1 に対して Method 4 は十分高速であり，プロトタイプ数が増加するほどその差は広がっていくことが確認できる．

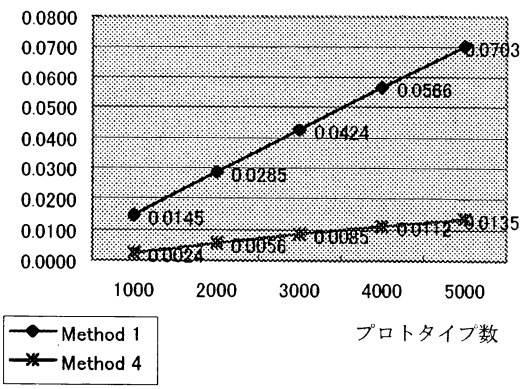

図9. プロトタイプ数の増加による実行時間の変化

次にプロトタイプのグループ化の効果を調べるために

・ Method 5: Method 4 +クラスタリングによるプロトタイプのグループ化

という手法と Method 4 との比較を行った．この結果は図 10 のようになった．

Method 5 のプロトタイプ数の増加による実行時間の変化は，Method 4 より緩和されたが，プロトタイプ数の増加に対して，実行時間を一定値に抑えることができておらず期待した結果とは言えない．

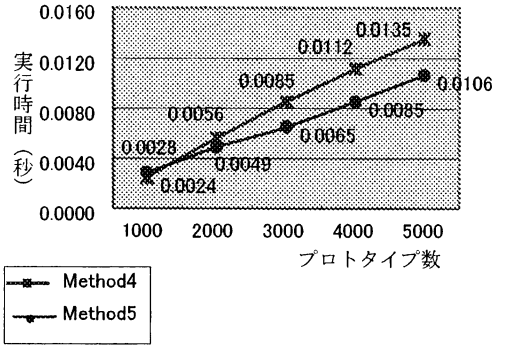

図１０．プロトタイプ数の増加に伴う実行時間の変化

5.3. 画像サイズによる実行時間の変化

提案手法では，画像の４分木表現を用いて階層的な評価を行うため，画像サイズへの依存が小さいと考えられる．そこで 600×480 の画像８枚（図 11）を２度ずつ回転させた各 180 枚，合計 1440 枚をプロトタイプとして登録し，Method 1 と

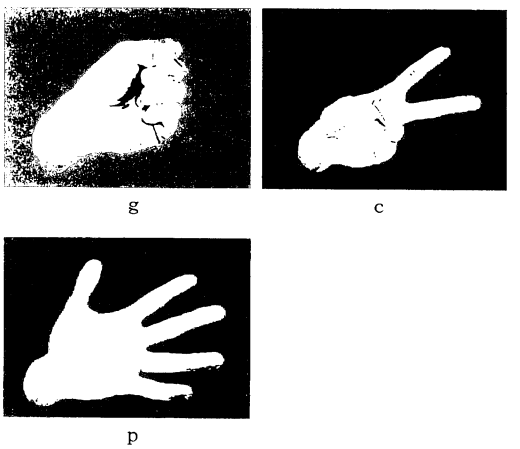

図１１．600×480 の画像

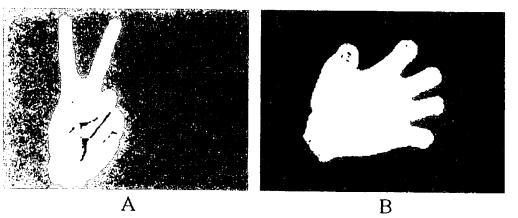

図１２．入力画像

Method 4の実験行った．Method 4では画像はレベル9の4分木にする．これは512×512の画像に相当するので，Method 1も画像サイズを512×512に正規化した．入力画像は図12のA，Bの2枚を用いて，その平均を実行結果とした．

	実行時間(秒)
Method 1	1.5497
Method 4	0.0357

表2．大きい画像サイズでの実行時間

表2は，その実行結果である．この実験では，提案手法を用いることにより，Method 4はMethod 1の約43倍高速化になった．

6. おわりに

本論文では，2値画像の4分木表現を用いることにより画像の最近傍識別を高速化する方法について検討し，画像の空間解像度を上げながら最近傍候補を絞り込むことができることを明らかにした．さらに，この絞り込みの効率を向上させるためのBest-First Searchを用いる方法，識別結果が確定した時点で探索を打ち切る方法を提案した．この方法は，直接画像間の距離計算を行う方法よりも高速であり，画像サイズが大きくなる程その差は大きくなる．512×512の画像1440個のプロトタイプを用いた最近傍識別実験では40倍以上の高速化が達成できることを確認した．

さらに，プロトタイプ間の関係を利用して，あらかじめプロトタイプを階層的にグループ化し，距離計算の初期の段階で入力と全プロトタイプとの距離計算を回避することにより，プロトタイプ数の増加に伴う実行時間の増加をある程度緩和することができた．しかし，この手法ではプロトタイプ数の増加に伴う実行時間の増加を一定値に抑えることができておらず根本的解決には至っていない．今後は，画像の4分木の階層表現と，クラスタの2分木の階層表現との関係を利用したより効率的なクラスタの絞り込み方法について検討し，さらに色ターゲット検出システム[8]に組み込み，領域の検出と識別を同時に行うシステムの開発を行う予定である．

文　献

[1] B.K.B T.M. Cover and P.E. Hart: Nearest neighbor pattern classification, IEEE Transactions on Information Theory, Vol.IT-13, No.1, pp.21-27, 1967.

[2] D. W. Aha: Case Based Learning Algorithms, Proc. of Case based Reasoning Workshop, pp.147-158, 1991.

[3] V. N. Vapnik: The Nature of Statistical Learning Theory, Splinger, 1995.

[4] Y. Freund and R. E. Schapire: A decision-theoretic generalization of on-line learning and an application to boosting, Journal of Computer and System Sciences, Vol. 55, No. 1, pp. 119-139, 1997.

[5] 和田，加藤``近接性グラフに基づく効率的Condensing の理論",信学技報 PRMU, Vol. 103, No. 96, pp. 13-18, May. 2003.

[6] 柴田，和田,加藤,"K-D Decision Tree: 最近傍識別器の高速化", 信学技報 PRMU, Vol. 103, No. 295, pp. 85-90, Sep. 2003.

[7] 武本，加藤，和田，"画像の4分木表現に対する最近傍識別"，信学技報 PRMU, Vol.103, No.390,pp.1-6,Oct.2003.

[8] 和田,"最近傍識別器を用いた色ターゲット検出－「らしさ」に基づかない識別とコンピュータビジョンへの応用－" 情処研報 CVIM134-3 pp.17-24, Sep. 2002.

[9] 工藤峰一；" 包含と排除によるK最近隣法の高速化"，信学技報 PRMU, Vol.103, No.295, pp.91 95,Sep.2003.

[10] A.Wojna: Center-Based Indexing for Nearest Neighbors Search, Third IEEE International Conference on Data Mining (ICDM2003), pp.681 684,Nov.2003.

[11] Murphy,~P.~M., & Aha,~D.~W. (1994).*UCI Repository of machine lernig databases* [http://www.ics.uci.edu/~mlearn/MLRepository.html].