社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

データクラスタリングによる局所特徴ベース情景内文字認識手法の改善

松田　崇宏†　　岩村　雅一†　　黄瀬　浩一†

† 大阪府立大学大学院工学研究科
〒 599–8531 大阪府堺市中区学園町 1–1
E-mail:　matsuda@m.cs.osakafu-u.ac.jp, {masa,kise}@cs.osakafu-u.ac.jp

あらまし　情景内文字認識とは，身の回りに存在するあらゆる文字を認識対象とし，それが何の文字であるかを認識することである．既存の情景内文字認識手法の一つである松田らの手法は，画像から得られる局所特徴の対応関係を利用することで，複雑な背景やレイアウトを持つ文字であっても，領域の検出と認識を同時に行なうことができる．しかし，登録されていないフォントに対しては十分な認識率を得ることが出来ない．この問題は大量のフォントをデータベースに登録することで解決できると考えられるが，その反面データベースや処理時間が増大してしまう．そこで本稿では，データベース中の類似する局所特徴をまとめるようなクラスタリングを行なうことで，登録するデータ数の削減を図る．データ数が減ることで，データベースと局所特徴の対応関係を求めるのに必要なコストを削減できると考えられる．そこで，クラスタリングを行なうことが認識率と処理時間にどのような影響を与えるかを調べるべく，実験を行なう．実験の結果，認識率を落とすこと無く，データベースと処理時間を大幅に削減できることがわかった．
キーワード　情景内文字認識，SIFT，クラスタリング，階層的 k-means 法

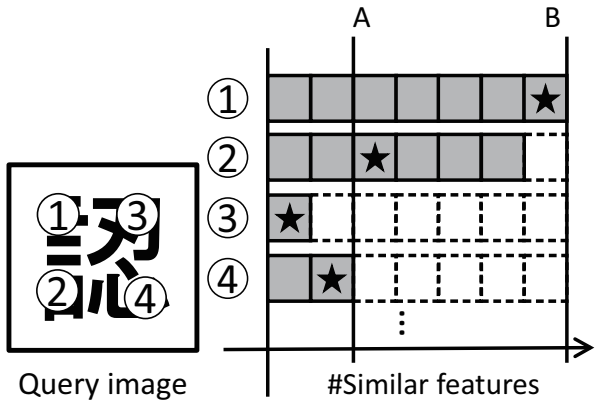
1. はじめに

情景内文字認識は，パターン認識の分野において注目を浴びているトピックの一つであり，様々な挑戦がなされている．一般に広く使われている文字認識（OCR）では，スキャナで撮像された文書に対しては高精度で認識できるが，カメラで撮影された情景内文字の認識は難しい．なぜなら，情景内文字の多くは制御された環境下で撮影されることは少なく，解像度や照明変化，回転，複雑な背景，ぼけ，遠近歪みなど，様々な環境的影響を受けているためである．そのような情景内文字を認識する試みは数多くなされているが，そのうちのほとんどがラテン文字のみを対象としている．また，文字認識手法の大半は，画像中のどこに文字があるかを判別する文字検出と，指定された文字が何であるかを識別する文字認識の，いずれかに焦点を当てている．そのため，これらを用いて実用的なものにするには，必ず手法を組み合わせなければならない．

これに対して松田らの手法 [1] は，ラテン文字に限らず全ての文字を認識対象とした情景内文字認識手法であり，照明変化に強く，複雑な背景やレイアウト上にある文字であっても認識することができる．また，事前に文字領域の切り出しを必要とせず，文字の検出と認識を同時に行うことができる．更に，解像度が 800×600 のクエリ画像に対しては，リアルタイム（標準的なパソコンで 1〜2FPS 程度）で動作する．この松田らの手法は局所特徴をベースとしており，局所特徴同士の対応関係を利用することで認識を行なう．

しかし，松田らの手法は事例ベース認識であり，あらかじめ認識対象と同じフォントの文字画像をデータベースに登録しておかなければならない．なぜなら，松田らの手法は基本的に同一のものを見つける手法であり，フォントの違いによって字形が大きく変わると，同一のものとみなされなくなるためである．これに対して，大量のフォントを登録すれば，未知のフォントであっても，類似の字形を持つ文字が登録されている可能性が高くなるため，フォントの違いに頑健になると考えられる．しかしその反面，登録した分だけデータ量が増大する．それと同時に，類似する特徴数も増加してしまう．例えば，漢字の「認」と「識」に共通する部首である「言」からは，ほぼ同一の特徴が得られる．同様に，フォントが違っても字形が似ていれば，それらから得られる特徴は類似する．松田らの手法では，クエリから得られるそれぞれの特徴に対し，それに類似する K 近傍の特徴をデータベースから探索し，それらの対応関係を利用して認識する．しかし，類似する特徴が複数存在する場合は，探索数を増やさなければならない．図 1 は，あるクエリ画像と，それから得られた特徴に類似する，データベース中の特徴点数の例である．例えば図 1(b) において，特徴点の探索を左から順に行なうとする．ここで，A まで探索を行なうと特徴 1 と特徴 2 しか正しい対応が得られない．全ての正しい対応を得るためには，B まで探索を行なう必要がある．しかし，全ての特徴点に対して同じ数の対応点を得なければならないため，類似しない特徴点（破線で囲われた分）まで得てしまう．これは冗長であり，処理時間の増加を引き起こす原因となる．そのため，大量のフォントをデータベースに入れることで，松田らの手法の利点の一つであるリアルタイム性が失われてしまう可能性がある．

そこで本稿では，類似する特徴を多く持つ特徴に対しては探



(a) (b)

図 1 (a) クエリ画像と得られた特徴点と，(b) その特徴に類似するデータベース中の特徴点数．★は，得られた特徴点に対応する特徴点を表す．

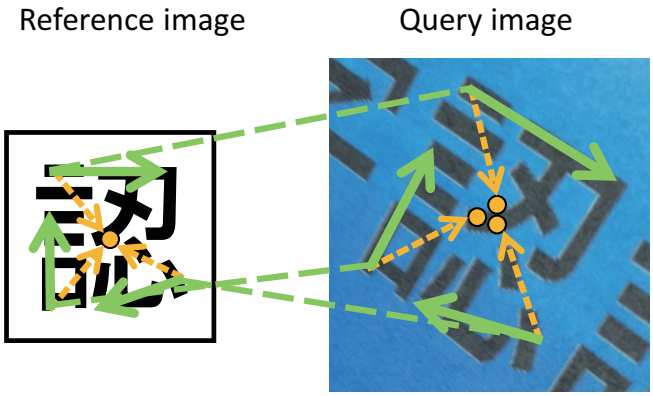


図 2 リファレンスポイントを求めるイメージ図．実線矢印が特徴，破線が特徴の対応関係，破線矢印が文字の中心（リファレンスポイント）の相対位置を表す．

索数を増やし，そうでない特徴に対しては探索数を減らすように，特徴毎に探索数を適応的に変化させることを目標とする．これを実現させるために，データベースに登録された類似する特徴同士をあらかじめクラスタリングでまとめておく．これにより，クラスタのセントロイドを探索するだけで，そのクラスタに属する複数の特徴を得ることができる．また，クラスタのサイズは類似する特徴点数によって変化するので，結果として特徴毎に対応点数を変化させることができる．このように，冗長な特徴や探索数を適応的に減らすことができるため，処理時間を減らすことができると考えられる．また，データベースにはクラスタのセントロイドのみを登録しておけば良いので，登録するデータ数を減らすことができると考えられる．今回はこれらを確認すべく，クラスタリングを行なうことが認識率や処理時間にどのような影響を与えるかを調査する．

2. 松田らの手法

松田らの手法 [1] は，大きく学習と認識の 2 つに分けることができる．学習時には，特徴抽出と近似最近傍探索手法の一つである Backet Distance Hashing (BDH) [2] の学習を行なう．BDH を使うことにより，近似最近傍探索を高速に行うことができる．認識時には，特徴抽出と特徴のマッチング，文字の領域検出と認識を行なう．本節では，松田らの手法の学習と認識のそれぞれについて説明する．

2.1 学　　習

2.1.1 特 徴 抽 出

学習に用いる文字画像を用意し，それら全てから SIFT 特徴 [3] を抽出する．SIFT 特徴は，その特徴が得られた座標，スケール，角度と，128 次元の特徴ベクトルを持つ．ここで，特徴点から構成されるデータベースを，特徴点データベースと呼ぶ．特徴点は，座標，スケール，角度の情報を持っている．この特徴点データベースは，文字の領域検出と認識時に利用される．また，128 次元の特徴ベクトルを特徴量と呼び，それのみで構成されるデータベースを特徴量データベースと呼ぶ．この特徴量データベースは，特徴のマッチングに利用される．

2.1.2 BDH のインデクシング

特徴量データベースに対して，BDH のインデクシングを行なう．事前にインデクシングを行なっておくことにより，特徴のマッチングを高速化できる．

2.2 認　　識

2.2.1 特 徴 抽 出

学習プロセスと同様に，クエリ画像から SIFT 特徴を抽出する．

2.2.2 特徴のマッチング

クエリから得られたそれぞれの局所特徴に対して，それに最も近い特徴ベクトルを持つものを特徴量データベースから探索する．そして，その特徴ベクトルに紐付けられた特徴点を特徴点データベースから取得し，それを対応点とする．

2.2.3 文字の領域検出と認識

クエリから得られた特徴点と，それに対応する特徴点データベースにある特徴点の対応関係を利用することで，文字の領域検出と認識を同時に行なう．まず図 2 のように，クエリから得られた特徴点に対し，その特徴点に対応するデータベースの特徴点から文字の中心への相対位置を用いることで，クエリ画像上の文字の中心を推定する．この時，クエリ画像中の文字は，スケールや回転により変化していることが考えられるため，特徴のスケールと角度の差分を考慮して文字の中心を推定する．このようなクエリ画像上に推定された文字の中心のことをリファレンスポイントと呼ぶ．正しく対応づいた特徴から得られるリファレンスポイントは文字の中心と等しくなり，それらは一点に集中する．つまり，集中していないリファレンスポイントの参照元の特徴点は正しく対応づいていないとみなすことができるので，そのような特徴点は早期に棄却することができる．文字の領域検出と認識を高速に行えるのは，この早期棄却を行なっているためである．そして，リファレンスポイントが集中している点に注目し，集中するリファレンスポイントの参照元であるクエリ上の特徴点の配置と，それに対応するデータベース上の特徴点の配置を比較する．これにより，文字領域の検出

と文字かどうかの判定を行なうことができる．

3. 局所特徴のクラスタリング

特徴量データベースには，類似する特徴ベクトルが数多く含まれている．そのため，クエリから得られた特徴に正しく対応する特徴を得るためには，類似する特徴全ての対応関係を求めなければならない．認識時にはリファレンスポイントや特徴点の配置の比較を行なっているため，誤認識を起こす可能性は低いが，処理時間の増加は避けられない．これは，特徴のマッチング時に全ての類似する特徴点を得るためには，K を大きくし，探索数を増やさなければならないからである．つまり，K が大きくなればなるほど，処理時間は増加する．また，文字の領域検出と認識時には，リファレンスポイントの計算や，厳密な特徴点の配置の比較にかかる時間も増加してしまう．そして，類似する特徴を多く持つ特徴だけではなく，類似する特徴をほとんど持たない特徴に対しても同数の対応点を求めなければならない．これは明らかに冗長である．

この問題を解決すべく，特徴ベクトルのクラスタリングを行なう．クラスタリングを行なうことにより，類似する特徴ベクトルを一つにまとめる．図 3 に，クラスタリング前のデータベースと，クラスタリング後のデータベースを示す．図 3 のように，クラスタリングは特徴量データベースに対してのみ行い，得られたクラスタのセントロイドを新たな特徴量データベースとする．これにより，クエリから得られた特徴の特徴ベクトルに最も近いセントロイドを探索することで，それに紐付けられた類似する特徴を複数得ることができる．つまり，類似する特徴ベクトルを多く持つ特徴は多くの対応点を，そうでない特徴は必要最低限の特徴のみを得ることができると考えられる．これにより，少ない探索回数でも多くの特徴を得ることができるため，処理時間を短縮することができると考えられる．また，セントロイドのみを探索対象にすれば良いため，登録する特徴ベクトル数が減り，近似最近傍探索の探索対象である特徴量データベースの削減が行なえる．

前述の目的のためには，類似する特徴のみが同じクラスタに含まれるようなクラスタリングが望ましい．ここで，類似する特徴が集まるということは，それらが特徴空間において散らばっておらず，密集しているということである．つまり，全てのクラスタの散布度をできるだけ小さくする必要がある．ここで散布度とは，データの散らばり具合を表す指標である．そこで，k-means 法を再帰的に行なう階層的 k-means 法 (例えば [4–6]) を用いる．クラスタ内の散布度が大きいクラスタに対してのみ再帰的にクラスタリングを行なうことで，全てのクラスタの散布度を小さくできると考えられる．しかし，[4–6] はそれぞれ方法が違い，いずれも目的にそぐわない．そこで今回は，クラスタリングを行なう毎に散布度を計算し，散布度が大きいものから順にクラスタリングする．また，特徴点数に対するクラスタ数の割合を C（$0 < C \leqq 1$）にすることを目標とするので，これを満たした時に処理を終了する．以下に，そのクラスタリング手法の手順を示す．

（1） 全ての学習用画像から得られた特徴ベクトルをサンプル集合 N とし，それに対して k-means 法を適用する．

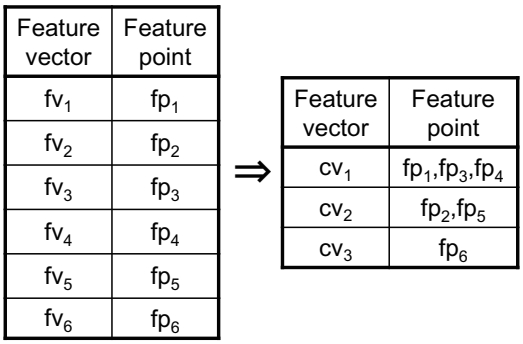


図 3 クラスタリング前とクラスタリング後のデータベースのイメージ図．左列が特徴量データベース，右列が特徴点データベース．

（2） クラスタ群から，次式で示される散布度が最大のクラスタを選択し，それをサンプル集合 $M \in N$ とする．

$$\sum_{i}^{|M|} ||s_i - c_{s_i}||^2 \tag{1}$$

s_i はサンプル集合 M におけるサンプル，c_{s_i} はサンプル s_i が属するクラスタのセントロイドを表す．

（3） サンプル集合 M に対して，k-means 法を適用する．

（4） クラスタ数が $|N|C$ 以上になるまで，(2) 以降を繰り返す．

4. 実験・考察

クラスタリング前の特徴点数に対するクラスタリング後の特徴点数（クラスタ数）の割合 C と，最近傍探索の探索数 K の値を変化させることで，認識率や処理時間がどのように変化するかを調べるべく，2 つの実験を行なう．以後，Prop. はクラスタリング有りの提案手法を，Conv. はクラスタリング無しの従来手法を指す．

認識性能の評価には，適合率と再現率，F 値を使用する．F 値は，適合率と再現率の総合的な評価の際に使用される評価基準であり，以下の式で表される．

$$F\text{ 値} = \frac{2 \cdot \text{再現率} \cdot \text{適合率}}{\text{再現率} + \text{適合率}} \tag{2}$$

4.1 実験条件

学習用画像には，MS ゴシックのひらがな，カタカナ，常用漢字，英数字の計 2458 文字の画像を用意した．クエリ画像には，[1] で用いたものを使用した．図 4 に例を示す．それぞれの画像は，ひらがな，カタカナまたは漢字のいずれかが 15 文字ず

表 1 実験に使用したパラメータ．seq(a,b,c) は，a から b まで c おきの値．例えば，seq(1,3,1) は 1，2，3 を表す．

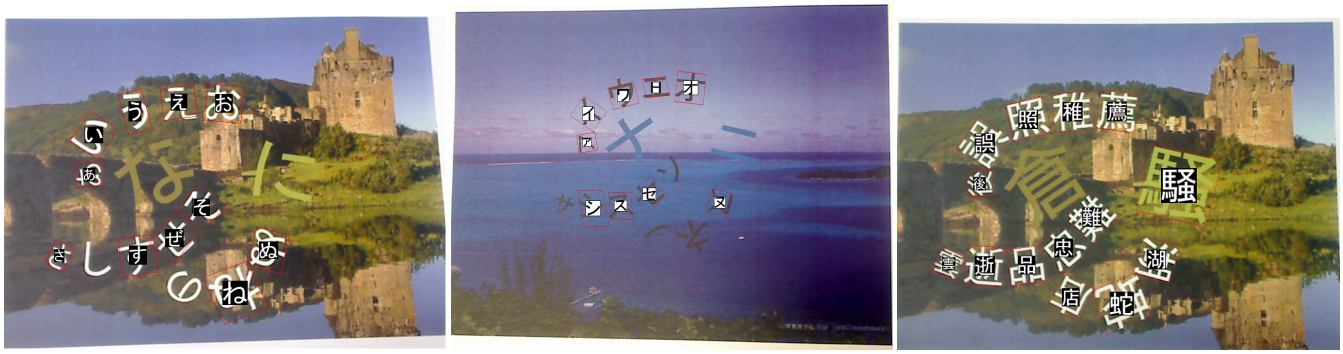
実験	従来手法	提案手法	
	K	C	K
1	seq(1,30,1)	seq(0.01,0.09,0.01) seq(0.1,0.9,0.1)	seq(1,30,1)
2	10	seq(0.1,0.9,0.1)	$C \times 10$
3	seq(1,30,1)	0.5	seq(1,30,1)

(a) (b) (c)

図 4　クエリ画像の一例

(a) (b) (c)

図 5　実験 2 における従来手法での認識結果（$K = 10$）.



(a) (b) (c)

図 6　実験 2 における提案手法での認識結果（$C = 0.5$, $K = 5$）.

つ書かれた紙を撮影したものである．これを，各字種 40 枚ずつ，計 120 枚用意した．文字はランダムに選択されており，全て MS ゴシックである．画像の解像度はいずれも 1600×1200 とした．使用した計算機の CPU は Intel の Xeon 3.3GHz，メモリは 512GB である．実験で用いたパラメータは，表 4.1 の通りである．

4.2 実　験　1

クラスタリングが認識率にどういった影響を与えるかを大まかに把握すべく，C と K の値を変化させて実験する．

結果を図 7〜9 に示す．図 7 は，C と認識率の関係を表す．これより，C が 0.1 から 0.9 の間では，クラスタリング有りと無しの場合でとりうる最高の F 値はほぼ同じであることがわかる．このことから，ある C に対して最適な K の値を用いることで，認識率を維持したままデータベースを削減できたと言える．しかし，C が 0.1 以下になると，認識率は徐々に下がる傾向にあることがわかる．これは，クラスタ数が少なすぎると，

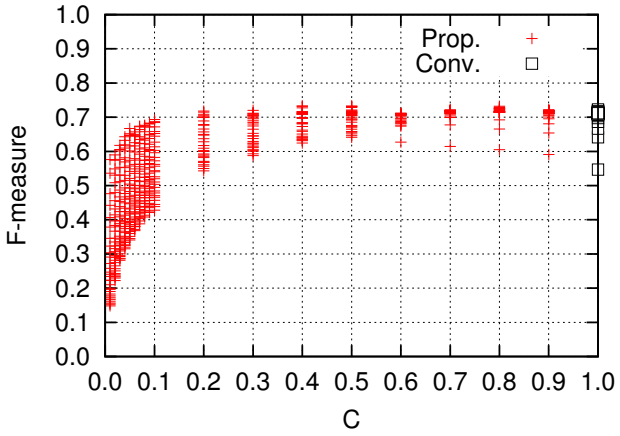


図 7　実験 1: C と認識率の関係.

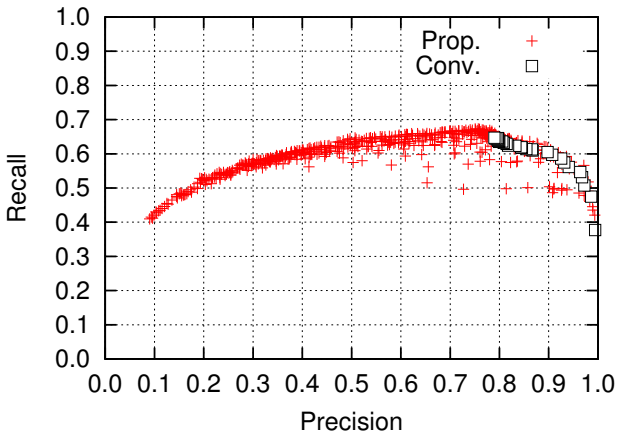


図 8　実験 1: 適合率と再現率の関係.

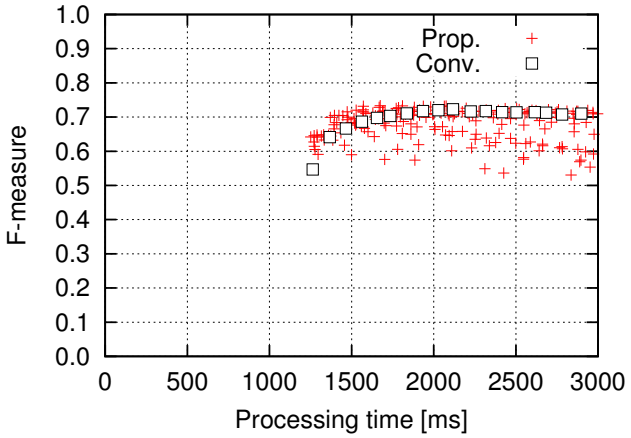


図 9　実験 1: 処理時間と認識率の関係.

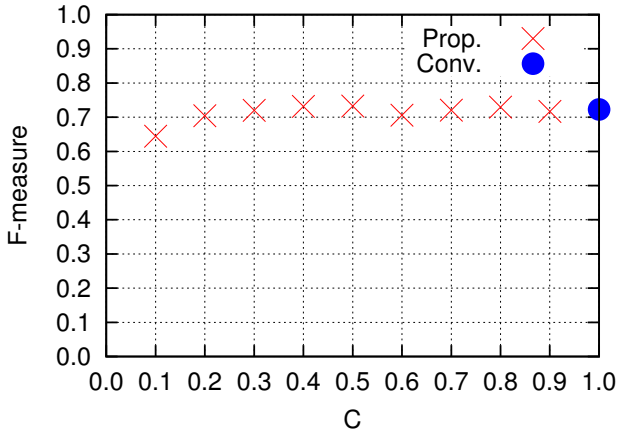


図 10　実験 2: C と認識率の関係.

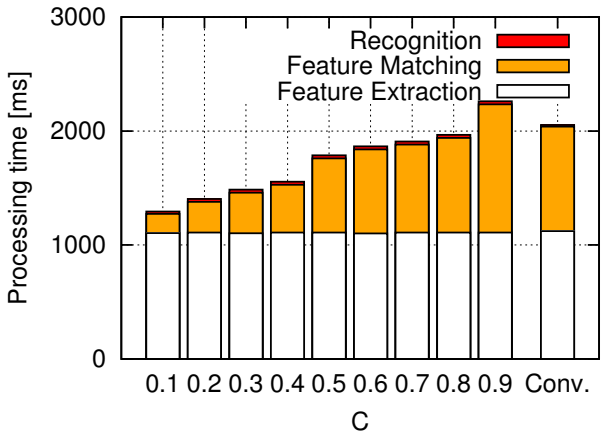


図 11　実験 2: C と各ステップの処理時間の関係.

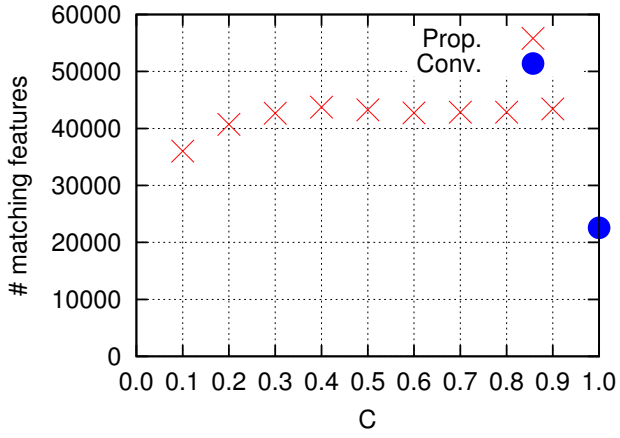


図 12　実験 2: C と対応点数の合計の関係.

クラスタには類似する特徴だけではなく，間違った特徴も含まれてしまうためである．ある程度の間違った特徴はリファレンスポイントや配置の比較により排除されるが，間違った特徴を過剰に含む場合には排除しきれず，誤認識を引き起こしてしまうと考えられる．認識率が低下したのは，これが原因であると考えられる．図 8 は，適合率と再現率の関係を表す．これより，クラスタリング無しの従来手法と同じような認識率と再現率を，クラスタリング有りの提案手法でも得ることができていると言える．図 9 は，処理時間と認識率の関係を表す．ここで処理時間は，認識の各処理（特徴抽出，特徴のマッチング，文字の領域検出と認識）の処理時間の合計である．両手法とも，処理時間が短い時ほど認識率が低くなっている．しかし，従来手法では 2100ms あたりで認識率が最も高くなるのに対して，提案手法では 1600ms あたりで最も高くなっている．また，従来手法では 1700ms あたりから認識率が 0.7 を下回り始めるのに対し，提案手法では 1300ms になるまで下回ることはない．これより，

クラスタリングを行なうことによって，認識率を落とすことなく処理時間を削減することができたと言える．

4.3 実　験　2

提案手法において，C と K の間には密接な関係がある．例えば，クエリから 100 の特徴点が得られたとする．クラスタリング無しの従来手法では，$K = 10$ とした場合，1000 の対応点が得られる．ここで，クラスタリング有りの提案手法でも同じ数の対応点を得る場合を考える．仮に $C = 0.5$ とすれば，特徴点に対応付いたクラスタには，平均 2 つの特徴ベクトルが含まれるため，K を 10 から半減させても同じ数の対応点数が期待できる．このように，クラスタに属するデータ数が増えれば増えるほど，K の値は小さくても十分な対応点数が得られることが期待できる．そこで，クラスタリング有りとクラスタリング無しの両方に関して，期待される対応点数を等しくすることで，認識率や処理時間にどのような影響を与えるかを調査する．

結果を図 10〜12 に示す．図 10 は，C と認識率の関係を表す．これより，期待される対応点数を同じにした状況においても，認識率を維持したままデータベースを削減できたことがわかる．また，結果画像の一部を図 5 と図 6 に示す．この結果画像を見ると，(a) の「い」や (a) の「イ」のように，クラスタリングによってシンプルな文字が認識できるようになっていることがわかる．しかし全体的には，従来手法で認識できるが提案手法で認識できないものや，その逆が多く見られたので，漠然とそのような傾向がありそうとしか言えない．また，図 5(c) と図 6(c) の比較からもわかるように，漢字に与える認識率の影響はほとんど確認できなかった．これは，十分な数の特徴が常に得られていたためであると考えられる．

図 11 は，C と処理時間の関係を表す．これより，データベースの削減と同時に，処理時間も削減できたと言える．これは，C が小さくなるに連れて K も小さくなるので，探索回数が減少してるためである．更に BDH の性質上，登録データ数が少なければ少ないほど，1 回の探索に要する時間が短くなることが知られている．つまり，登録する特徴ベクトル数が減ったことで，BDH による 1 回の探索に要する時間が減少したことも考えられる．また，文字の領域検出と認識にかかる処理時間の全体に占める割合は非常に小さいことがわかる．これは，認識に寄与しない特徴点を，リファレンスポイントを使用することで早期棄却をしているため，高速に処理できるためである．一方で，$C = 0.9$ の時は従来手法よりも処理時間が長くなっている．そこで，図 12 に注目する．

図 12 は，C と，1 枚あたりから得られる合計対応点数の関係を表したグラフである．これより，クラスタリング後の対応点数は，クラスタリング前の対応点数に比べて多くなっていることがわかる．$C = 0.9$ の時に従来手法よりも処理時間が長くなっているのは，類似する多くの特徴を得る際に生じたオーバーヘッドのためであると考えられる．ここで，期待される対応点数を同じにしたのに，なぜ実際に得られた対応点数が倍近くあるのかについて考えてみる．もしクラスタが持つ特徴点数に偏りが生じていないのであれば，得られる対応点数は等しくなるはずである．また偏りが生じていたとしても，クラスタがランダムに選択されているのであれば，対応点数は等しくなるはずである．これらから，クラスタが持つ特徴点数には大きな偏りが生じており，かつ認識に利用される特徴点は類似した特徴を多く持つことが考えられる．図 1(b) で言えば，1 や 2 のような特徴が認識に多く寄与していることになる．しかし，クラスタリング無しの従来手法では，半分の対応点数で同じ認識率を達成している．これから，提案手法では多くの類似する特徴を得ることができているが，それらの多くは冗長な特徴である可能性がある．

クラスタリングによって，対応点数を特徴毎に動的に変化させることで，処理時間とメモリ使用量は大きく削減できた．一方で，クラスタリングをしない場合に比べて，冗長な特徴が増加している可能性が示唆されたため，更なる検討が必要である．

5. まとめと今後の課題

今回は松田らの手法において，対応点数を適応的に変化させるべく，特徴のクラスタリングを行なった．そして，クラスタリングが性能にどのような影響を与えるかを調査した．結果として，認識率を維持したまま，データベースのサイズと処理時間を削減することができた．

今回は，1 フォントのみを用いて実験を行なった．今後の課題としては，複数のフォントをデータベースに登録しクラスタリングをすることで，性能にどのような影響を与えるかを調査することが挙げられる．

謝辞　本研究は， JST CREST の補助による．

文　献

[1] T. Matsuda, M. Iwamura, and K. Kise, “Performance improvement in local feature based camera-captured character recognition,” Proceedings of the 11th IAPR International Workshop on Document Analysis Systems (DAS2014), pp.196–201, April 2014.

[2] M. Iwamura, T. Sato, and K. Kise, “What is the most efficient way to select nearest neighbor candidates for fast approximate nearest neighbor search?,” Proc. 14th International Conference on Computer Vision (ICCV 2013), pp.3535–3542, Dec. 2013.

[3] D.G. Lowe, “Distinctive image features from scale-invariant keypoints,” IJCV, vol.60, no.2, pp.91–110, 2004.

[4] D. Nistér and H. Stewénius, “Scalable recognition with a vocabulary tree,” IEEE Conference on Computer Vision and Pattern Recognition (CVPR), vol.2, pp.2161–2168, June 2006.

[5] K. Arai and A.R. Barakbah, “Hierarchical k-means: an algorithm for centroids initialization for k-means,” Reports of the Faculty of Science and Engineering, vol.36, pp.25–31, 2007.

[6] B. Chen, J. He, S. Pellicer, and Y. Pan, “Using hybrid hierarchical k-means (hhk) clustering algorithm for protein sequence motif super-rule-tree (srt) structure construction,” International journal of data mining and bioinformatics, vol.4, pp.316–330, Inderscience, 2010.