カメラで撮影した単語画像の実時間認識

岩村　雅一†　　辻　　智彦†　　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: {masa,kise}@cs.osakafu-u.ac.jp

あらまし　本稿ではカメラで撮影した単語画像の実時間認識手法を提案する．提案手法を用いることで，環境中の単語を認識し，あらかじめ単語に関連付けられた情報（画像，音声，翻訳など）を利用者に提供する情報提示システムが実現できる．このシステムはノートパソコンで実時間で動作し，斜めから撮影した文字列も認識可能である．また，文字が曲線状に配置されていても，単語内で文字の大きさが変化しても認識可能である．

キーワード　カメラベース単語認識，実時間処理，事例，近似最近傍探索

Real-Time Recognition of Camera-Captured Word Images

Masakazu IWAMURA†, Tomohiko TSUJI†, and Koichi KISE†

† Graduate School of Engineering, Osaka Prefecture University
1-1 Gakuencho, Naka, Sakai, Osaka, 599-8531 Japan
E-mail: {masa,kise}@cs.osakafu-u.ac.jp

Abstract　In this report, we propose a real-time recognition method of camera-captured word images. Using the method, we can realize a system which recognizes words in scenes and offers related information to the word such as images, sounds and translation of the words in real-time. The system can recognize character images captured from various angles, texts aligned on a non-straight line, and texts which consists of characters of various sizes.

Key words　Camera-Based Word Recognition, Real-Time Processing, Case Example, Approximate Nearest Neighbor Search

図 1: 提案手法を用いる情報提示システム．添付資料にデモビデオあり．

1. はじめに

本稿ではカメラで撮影した単語画像の実時間認識手法を提案する．認識対象は印刷文字である．提案手法について述べる前に，提案手法を用いることで実現した情報提示システムを図 1 に示し，概要を述べる．このシステムは小型カメラで撮影した環境中の単語を認識し，あらかじめ単語に関連付けられた情報（画像，音声，翻訳など）を利用者に提供することができる．このシステムはノートパソコンで実時間で動作し，斜めから撮影した文字列も認識可能である．また，図中の文書のように文字が曲線状に配置されていても，単語内で文字の大きさが変化しても認識可能である．これは文字行が平行に並んでいることを仮定した従来手法 [1] では実現できない．

提案手法は図 1 のような複雑なレイアウトの文書に対応するために，最初に文字単位の認識を行い，その結果を受けて単語を認識するという二段階の処理を行う．

文字認識処理ではカメラで撮像した印刷文字を認識対象とするため，撮像した文字画像に生じる射影歪みや解像度の低下，ぼけといった画像の劣化に対処して実時間で認識する必要がある．これらの要求を満足するために，本稿では文献 [2] のカメラベース文字認識手法を出発点とする．この手法は，クラスラベル付きの文字画像データ（事例）をデータベースに蓄積しておき，事例に基いて文字を認識する．この手法は実時間で動作するものの，データベースに多数のフォントを登録すると認識性能が低下する問題があった．そこで本稿では文献 [3] で提案されている高速かつ頑健な特定物体認識のための近似最近

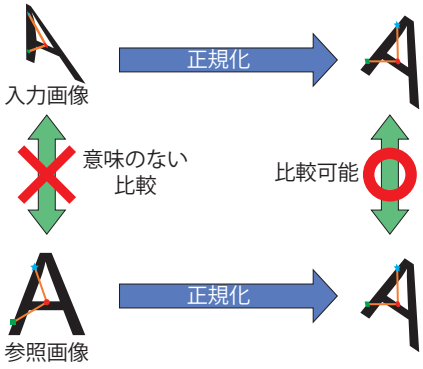


図 2: ベース手法 [2] における画像（連結成分）の照合方法.

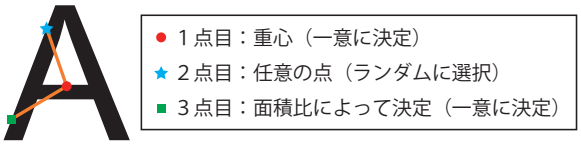


図 3: ベース手法 [2] における対応する 3 点の選択方法.

傍探索手法を新たに導入することでこの問題を解決する.

単語認識処理では，文字認識処理で推定した文字の向きのみを考慮して単語を決定する手法を提案する．上記の文字認識処理は射影歪みに頑健であるため，射影歪みを受けると類似してしまう字種（「p」と「q」など）の区別が困難である．そこで，文字認識処理では字種の詳細な認識を避け，単語の決定と同時に字種も決定する.

以下では簡単のために，白色の背景に黒色の文字が書かれていることを想定する．文字の連結成分 (注1) は簡単な処理で切り出せるとする．また，全ての文字は同一平面上に存在するとする.

2. 文字認識の従来手法

本節では文献 [2] の認識手法（以下，ベース手法）の概要を述べる．その核となるのは，図 2 に示すアフィン変換を受けた連結成分の照合部分である．通常，画像に歪みがある場合，2 画像を重ね合わせて単純に比較することはできない．しかし，もし画像中の対応する 3 点が既知であれば，点を結ぶ線分が直交し，長さが同じになるように画像を正規化することで比較可能となる．このような対応する特徴点に基づく画像照合手法としては，Geometric Hashing (GH) [4], [5] がよく知られている．GH では多数の特徴点から 3 点の組み合わせをランダムに選択し，図形の照合を試みる．この処理に要する計算量は，特徴点数を P とすると $O(P^4)$ である．ベース手法では照合する対象が連結成分であることを利用し，特徴点を図 3 のように定めることで $O(P^2)$ に削減した.

ベース手法は図 4 のように分離文字取扱部，登録部，認識部で構成されており，以下では各部の処理について概説する．なお，パラメータに関する記述は本稿の実験で使用した値であり，文献 [2] と同一とは限らない.

(注1)：画像中で黒画素が隣接し，一塊になっているものを指す.

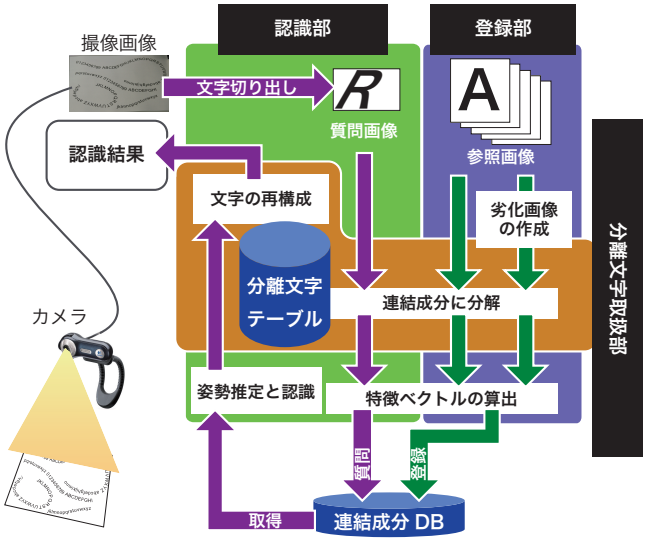
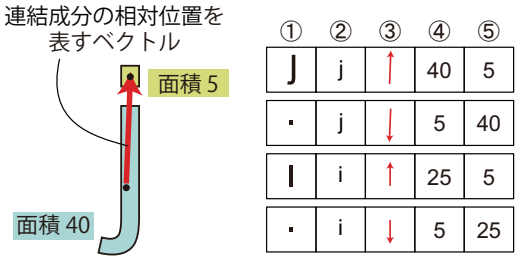


図 4: 文献 [2] の認識システムの概要.



(a) 連結成分の記述方法　(b) 分離文字テーブル

図 5: 分離文字の記述方法．分離文字テーブルの要素は左から，①連結成分の形状 (連結成分番号)，②元の文字，③連結成分の相対位置，④自連結成分の面積，⑤組になるべき連結成分の面積である.

2.1 分離文字取扱部

本節では「i」や「j」等，複数の連結成分から成る文字（分離文字）の取り扱いについて述べる．分離文字を扱うために，登録時に文字を構成する連結成分の数を調べ，2 つ以上あれば，図 5 に示す分離文字テーブルに 2 つの連結成分の相対位置や大きさの関係を記載する．認識時に分離文字テーブルを参照し，この条件を満たす連結成分があれば連結成分を統合し，一文字と認識する.

Arial フォントの場合,「i」の下部,「I（アイ）」,「l（エル）」はアフィン歪みを受けると外見が同一になり，識別不能である.「i」を正しく認識するためには,「I」や「l」のように外見が同一である連結成分も「i」の一部であるか調べる必要がある．そのため，あらかじめクラス分け処理を行い，アフィン変換によって同一形状になる連結成分を同一クラスとみなす．これは，参照画像を登録する際に，作成中のデータベースを用いて 1 枚ずつ認識し，既に類似の連結成分が登録されていれば同じクラスに統合する処理である．その際，クラス分けによって異なる字種の連結成分が同一クラスになってしまう場合があることに注意が必要である．図 6 に示す例では，劣化

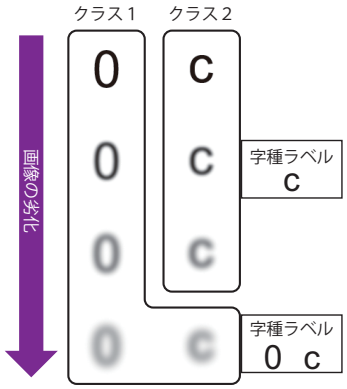


図 6: 連結成分のクラス分け処理の例.

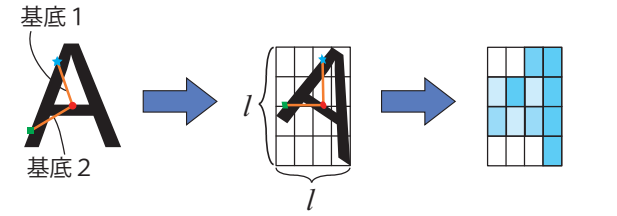


図 7: 特徴ベクトルの計算方法.

した「c」が「0」と同じクラス 1 に属しているため，認識時にクラス 1 であると認識された連結成分が「0」か「c」かを区別できない．このような同一クラス内の字種の判別は後段の単語認識処理に委ねる．

2.2 連結成分登録部

ここでは二値化した連結成分の画像をアフィン不変な特徴ベクトルで記述し，データベースに登録する．

2.2.1 劣化画像の生成

撮影時のピンぼけや解像度の低下に対処するため，生成型学習法 [6] を用いて劣化した参照画像を生成する．本稿では，ガウスぼかしの重畳を 3 段階 (適用なしを含む) と解像度の低下を 3 段階 (適用なしを含む) を組み合わせた合計 9 段階で行い，元の参照画像を含めて 9 枚の参照画像を得る．生成された劣化画像は再び二値化し，以後は元の参照画像と同様に扱う．

2.2.2 特徴ベクトルの生成

特徴ベクトルを以下の二段階の処理によって計算する．

一段目では，図 7 に示すように，2 本の基底によって張られる不変座標系を用いて k 次元の特徴ベクトルを作成する．図 7(a) を 2 本の基底を用いて正規化したものが図 7(b) である．そして，$k(=l\times l)$ 個の均一な部分領域を設定し，図 7(c) のような黒画素数のヒストグラムを計算する．最後にヒストグラムの合計が 1 になるように正規化し，k 次元の特徴ベクトルを得る．予備実験の結果から，本稿では $l=4$ とした．

二段目では，$(3k)$ 次元の特徴ベクトルを得る．図 7(a) のように，特徴点が 3 点あれば，そのうち 1 点を原点とすることで 2 本の基底ベクトルが計算できる，原点の取り方は 3 通りあり，基底ベクトルの取り方も 3 通りあるため，上記の方法で 3 本の k 次元特徴ベクトルが計算できる．これを単純に結合することで得られる $3k$ 次元の特徴ベクトルを提案手法で使用する．結合の順序は特徴点が選択された順序に基づいて一意に決定する．理論上，同一の 3 点が得られる限り，アフィン歪みの影響を受けずに常に同じベクトルが計算できる．

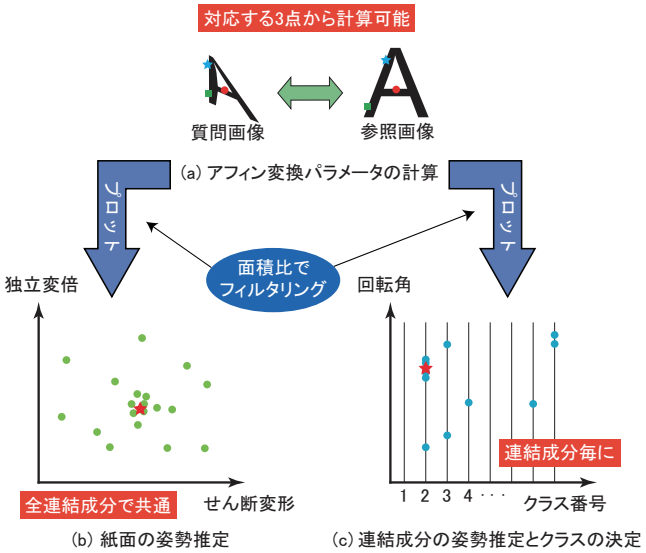


図 8: 姿勢推定を伴う認識.

2.2.3 連結成分データベースへの登録

ハッシュ表に連結成分の情報を登録する．連結成分毎に保持する情報は，クラス番号，特徴ベクトル，基底の作成に用いた特徴点 3 点の座標である．衝突が起こる場合はリストで連結する．特徴点の座標を登録しておくのは，認識の際の姿勢推定で利用するためである．

ハッシュ値 H_{index} は次式で計算する．

$$H_{\mathrm{index}}=\left(\sum_{i=1}^{3k}D^{i-1}r_i\right)\bmod H_{\mathrm{size}},\qquad(1)$$

ここで H_{size} はハッシュ表の大きさであり，$H_{\mathrm{size}}=2^{19}-1$ とした．r_i は特徴ベクトルの i 番目の要素を D 段階に量子化した値であり，$D=2$ とした．衝突が起こる場合はリストで連結する．

2.3 連結成分認識部

2.3.1 画像の取得と連結成分の切り出し

画像はデジタルカメラや web カメラで静止画ないし動画として取得する．動画として撮像した場合はフレーム毎に分解して，複数の静止画として扱う．得られた各画像を質問画像と呼ぶ．質問画像に適応二値化と輪郭抽出手法を適用し，連結成分を抽出する．

2.3.2 特徴ベクトルの生成

連結成分から特徴ベクトル（以後，クエリ特徴ベクトルと呼ぶ）を生成する．この処理は 2.2.2 の処理と基本的に同じである．唯一異なるのは，速度向上のために，

作成可能な全ての特徴ベクトルを計算するのではなく，あらかじめ定める S 個に限定することである．S が小さいと認識率は低下するものの処理時間は小さくなる．本稿では $S = 10$ とした．

2.3.3 認識と姿勢推定

連結成分の認識と同時に姿勢を推定する．

最初に紙面の姿勢を推定する．本稿では全ての文字は同一平面 (紙面) 上に存在すると仮定している．この場合，図 8(a) のように特徴点の対応関係から求めた 4 つのアフィン変換パラメータのうち，せん断変形と独立変倍のパラメータは全ての連結成分で共通になるはずである．そこでベース手法では図 8(b) のような 2 次元空間での密度推定を利用して，尤もらしいパラメータの組を推定する．具体的には，プロットされた点の中から近傍の密度が最も高い点 (図 8(b) の星マーク) を選択する．推定には質問画像の全連結成分から推定したアフィン変換パラメータを使用する．ただし，これらの情報には誤ったものが含まれるため，詳細は省略するが，投票や閾値処理等によって信頼できないものを除外する．

次に，連結成分毎に認識結果を定める．図 8(c) のような連結成分の回転角とクラス番号が作る 2 次元空間での密度推定を利用して，尤もらしい回転角のパラメータとクラス番号の組を推定する．図 8(b) の場合と異なるのは，回転角は連続量であるが，クラス番号は離散であることである．そのため，各クラス番号について 1 次元の密度推定を行う．以上の処理により，各連結成分の種類 (クラス番号) と姿勢 (せん断変形，独立変倍，回転角) を求めることができる．

3. 文字認識の提案手法

本節では，前節で述べたベース手法に文献 [3] の方策を 3 つ導入することにより，文字認識処理を改良する．

1 番目の方策は距離計算の導入である．前節で述べたように，ハッシュ表から得られる情報には誤りが含まれるため，その中から正しいものを選択する必要がある．ベース手法では投票や閾値処理等で信頼性の高い情報を絞り込んだが，提案手法では代わりにクエリ特徴ベクトルと，ハッシュ表から得られた特徴ベクトルのユークリッド距離を計算し，距離が閾値以下のものを選択する (注2)．

2 番目の方策では，図 9 に示すように，クエリ特徴ベクトルのビット反転によって新たなクエリ特徴ベクトルを作成する．本稿では，48 次元の特徴ベクトルに対して $e = 0.002$ と $b = 8$ を用いた．

3 番目の方策はハッシュ値の衝突に関する工夫である．ベース手法ではハッシュ表の一部のビンで大量の衝突が起こることがあった．ハッシュの処理時間は衝突の数に比例するため，衝突が大量に発生すると処理時間が非常

(注2)：文献 [3] の手法は距離が最小のもののみを選択し，本稿の手法と若干異なる．予備実験のおいて両者を比べたところ，閾値を用いる方が性能が良かった．

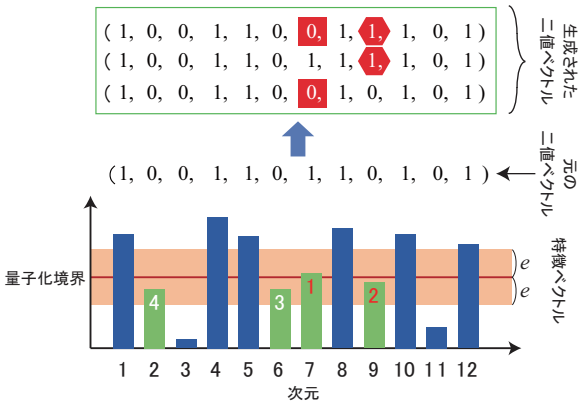


図 9: 文献 [3] のクエリ特徴ベクトルの二値化方法．この例では，特徴ベクトルは 12 次元である，実数値ベクトルの二値化の際に，閾値とベクトルの各要素の差を昇順に並べ，差が e より小さい要素を最大 b 個（この例では $b = 2$）選択し，ビットを反転させる．選択の際には差が小さい要素を優先する．図の例では，7 次元目と 9 次元目が選ばれ，ビット反転され，3 本の新しいベクトルが生成された．

に遅くなるという問題があった．そこで本稿では，衝突の数が c より大きくなったビンは c 個に間引きくこととした．すなわち，衝突が大量に発生したビンの要素を c 個を除いて削除する (注3)．これにより，メモリ使用量と処理時間を大幅に削減できる．本稿では $c = 20$ とした．

4. 単語認識手法

本稿の単語認識処理について述べる．この処理では，通常行われるように単語辞書を使用せず，文字の並びの尤もらしさのみを基準として単語を決定する．本稿では英語のように単語間がスペースで区切られ，かつ左から右に記される言語を想定する．前提条件として文書画像は文字認識の段階で独立変倍とシアーの歪みが取り除かれたものとする．「文字の向き」とは文字認識処理で求められる回転のパラメータを指す．

最初に単語領域を推定する．図 10 のように文書画像をぼかして二値化することで単語領域を推定する．適切なぼかしの度合いはキャプチャ画像中の文字の間隔と太さで決まるため，文字間の平均距離を d，連結成分の平均画素数を a としたとき，標準偏差 $\sigma = 200\ d/a$ のガウシアンフィルタを用いてぼかした．

そして各単語を構成する文字を決定する．図 10 を例に挙げると，単語領域 2 から抽出される文字は ①〜⑤ である．この段階での「文字」は文字認識処理によってクラスレベルでの識別が終了した状態であり，未だ文字毎に複数の字種の候補を持つ．今回の説明では，文字 ①

(注3)：文献 [3] の手法は閾値以上の衝突が起こると，そのビンに登録されている情報を全て削除するため，本稿の手法と若干異なる．文献 [3] の方法をそのまま試したところ，「O」のような円形の字種だけが選択的に認識できなくなった．この原因として，円形の文字ではほぼ全ての特徴ベクトルが同じビンに登録され，そのビンの情報が全て削除されると全く認識できなくなることが考えられる．

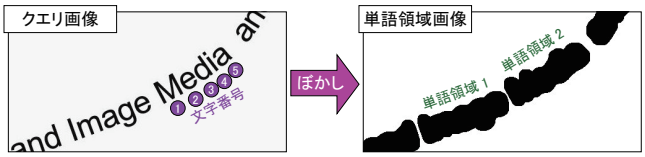


図 10: 単語領域の取得.

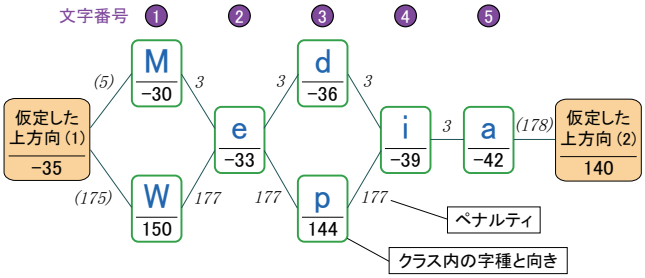
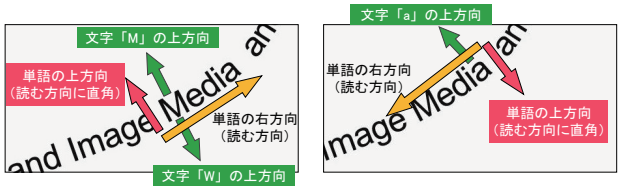


図 11: 文字の探索とペナルティの計算.



(a) 文字 ① から読む場合. (b) 文字 ⑤ から読む場合.

図 12: 上方向の仮定.

Arial 0123ABCDabcd
Century 0123ABCDabcd
Times New Roman 0123ABCDabcd
Verdana 0123ABCDabcd
Meiryo bold 0123ABCDabcd
OCRB 0123ABCDabcd
Book Antiqua Bold 0123ABCDabcd
BellGothicStd-Black 0123ABCDabcd
Franklin Gothic Medium 0123ABCDabcd
TektonPro-Bold 0123ABCDabcd

図 13: 認識に用いたフォントの一部（1〜10 番目のフォント）.

のクラスには「M」と「W」の 2 つ，文字 ③ のクラスには「d」と「p」の 2 つの字種を含み，それ以外は 1 字種のみで構成されたクラスであるとする．

提案手法では文字の並びや向きが急激に変化することはないと仮定して単語内の文字の並びと字種を推定する．文字の並び順については，図 10 の単語領域 2 の単語を ④⑤①②③ と途中で急に離れた場所の文字を読むようなケースはまず考えられない．そこで単語内の文字を最短で 1 度ずつ通るものを選択する．結果として ①②③④⑤ およびその逆順 ⑤④③②① が得られる．

次に，文字の向きに関する 2 種類のペナルティを設けて尤もらしい単語候補を求める．ペナルティの計算には文字認識段階で得られる文字の回転角を利用する．回転角はキャプチャ画面の上方向を 0 度として時計回りの回転で回転角が大きくなるとする．図 11 は推定された並び順に単語領域 2 の各文字の候補を挙げたグラフである．図中の各字種の下の数値が回転角を表す．

1 つ目のペナルティとして，「文字の向きは急激に変化しない」という仮定に基づき，1 つ前の文字との回転角の差を用いる．角度差の定義域は 0° ～ 180° である．例えば図 11 において「e」から「d」に進んだ時，2 つの文字の向きは 3° 違うため，ペナルティとして 3 を加算する．経路の中には何度も大きなペナルティが加算される組み合わせが存在する．その場合は途中で計算を打ち切り，候補から除外することで処理時間の増加を抑える．

2 つ目のペナルティでは，「単語は左から右に読む」というルールに基き，単語の先頭の文字の向きの妥当性を検証する．これにより，文字の並び順（「Media」か「aideM」か）を決定できる．単語領域 2 の単語を例に取り，考え方を図 12 で示す．図 12(a) のように単語を文字番号 ① から順に黄色の矢印の方向に読むと，単語の上方向（赤色の矢印）と先頭の文字が「M」であるときの上方向がほぼ一致する．この角度の差をペナルティとする．すなわち，これは図 11 における「仮定した上方向 (1)」および次ノードとの間のペナルティに相当し，ペナルティは 5 となる．もし先頭の文字が「W」であれば 175 という大きなペナルティが可算される．また，図 12(b) のように単語を ⑤ から順に読む場合も角度の差が大きくなり，178 という大きなペナルティが可算される．

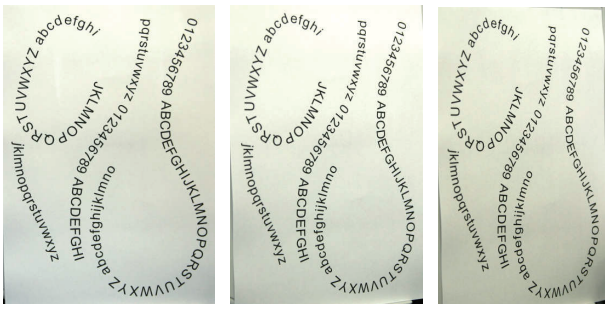
以上のように文字候補の組み合わせごとにペナルティの合計を求めた後，昇順にソートすることで尤もらしい文書中の単語を推定する．例で用いた単語領域 2 の場合，「Media」がペナルティ 17 で最小となる．1 文字ごとの認識では区別できなかった「d」と「p」のような同一クラスの字種も単語認識の段階で文字レベルで区別されるようになった．しかし，「0」と「O」のように互いに拡大縮小の関係にあり，向きが類似する字種についてはペナルティがほぼ等しくなり，尤もらしい字種の決定が困難である．この問題に対する改善方法としては，単語辞書を使用すること等が考えられる．

5. 実　　験

提案手法の有効性を確認するために，カメラで撮像した文字や単語を認識した．文字認識についてはクラスレベルの認識であり，字種の区別までは行わない．また，単語認識では文字認識の提案手法とのみ組み合わせた．

文字認識については最大 100 フォント，単語認識については最大 10 フォントを登録したデータベースを使用した．使用したフォントは，Microsoft Windows 7 にインストールされているフォントから選択した 100 種類である．そのうち 10 フォントを図 13 に示す．選択の際，線幅の細いフォントは解像度低下等の影響によって線が途切れ，連結成分が分解され易いため，除外した．

実験に使用した字種は，大文字と小文字のアルファベットと数字の合計 62 字種である．生成型学習により，1 字種当たり 9 種類（劣化なし 1 種類を含む）の劣化画



(a) 0 度. (b) 30 度. (c) 45 度.

図 14: 文字認識の対象（Arial）

像を生成したため，10 フォントでは合計 5,580 枚，100 フォントでは合計 55,800 枚の参照画像をデータベースに登録した．計算機は CPU が Opteron 2.8GHz，メモリが 32GB のものを使用した．画像の登録と認識に要する計算量を削減するため，連結成分の幅と高さの大きい方が，参照画像では 100 ピクセル，質問画像では 50 ピクセルになるように正規化した．また，前節で述べた方策 1 の距離の閾値を t/r と定めた．ここで r は前述の正規化後の連結成分の黒画素数である．t は定数で，登録部のクラス分類時には $t = 200$，認識部には $t = 80$ を用いた．

5.1 文字認識実験

文字認識の対象として，図 14 のような画像を用いた．これらは文字が曲線上に並ぶようにレイアウトされた文書で，それぞれ 1 フォントのみからなる．1 枚につき 1 フォントが使用されている．各文書は 62 字種を 2 回ずつ，合計 124 文字を含む．文書を A4 用紙に印刷し，デジタルカメラで正面に対し 0 度，30 度，45 度から撮像したものを図 14 のように手動で切り出した．図 14(a) の画像の大きさは 1577×2209 画素であった．

実験では，登録フォント数を 1〜100 まで徐々に増加させ，認識率と処理時間の変化を見た．登録フォント数は，1〜10 フォントまでは 1 フォントずつ，それ以降は 5 フォントずつ増加させた．認識対象はデータベースと同じフォントのものを用いた．すなわち，1 フォント目は Arial をデータベースに登録し，Arial の文書画像を認識した．2 フォント目は Arial と Century を登録し，同じく Arial と Century の文字画像を認識した．ただし，認識対象を 75 フォント分しか用意できなかったため，75 フォント以降は 75 フォントを認識対象として使用した．

2.1 で述べたように，連結成分は登録処理の際に自動的にクラス分けした．その際，提案手法で述べた 2 番目の方策（クエリ特徴ベクトルを生成する方策）は使用しなかった．クラス分けの様子を示す例として，Arial の 62 字種（9 種類の劣化画像を用いるので 558 画像）をクラス分けした結果を表 1 に示す．表には生成された 55 クラスのうち，2 字種以上が属するクラスのみを列挙した．

表 1: Arial の 62 字種から自動的に生成された 55 クラスのうち，2 字種以上が属するクラス．

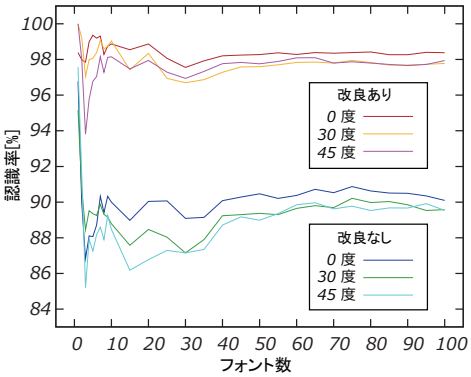
0 O o	6 9	7 L	C c	E m	I l	N Z z
S s	V v	W w	b q	d p	n u	

2. で述べた文献 [2] のベース手法（「改良なし」）と 3. で述べた改良手法（「改良あり」）の文字（クラス）認識結果を比較した．結果を図 15 に示す．まず，図 15(a) は文字認識率である．ベース手法は複数のフォントを登録すると認識率が減少したのに対し，提案手法は登録フォント数に依らず，高い水準でほぼ一定であった．100 フォントの場合，提案手法では正面から撮像した文字の認識率が 98.4%であり，ベース手法に比べて約 8%の向上を達成した．また，45 度から撮像した場合でも 97.9%の認識率が得られた．なお，提案手法での認識失敗の例としては誤認識の他，図 15(b) の「QR」と「VW」のように，隣接した文字間で連結成分が統合されてしまい，認識が不可能となったことが挙げられる．

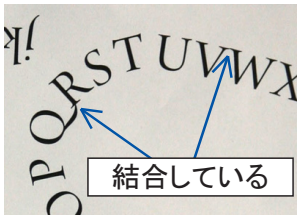
次に，図 15(c) に 1 文字当たりの平均処理時間を示す．両手法とも，登録フォント数の増加に伴って処理時間が増加したが，提案手法では増加の傾きが緩やかになった．100 フォント，0 度の場合，提案手法による処理時間は 7.2ms であり，ベース手法より約 70%の時間削減となった．以上より，提案手法で導入した 3 方策が有効であることが確認できた．

ここで，導入した 3 方策がそれぞれ文字認識性能にどのような影響をもたらしたのかを考察する．図 15(d) に方策の組み合わせと認識性能の関係を示す．フォント数は 100，カメラ撮影角は 0 度である．方策 1 を適用することで，認識率の向上と処理時間の削減の効果が得られた．これは，認識と姿勢推定時に用いるアフィン変換パラメータから誤ったものが取り除かれたためと考えられる．方策 2 は認識率の微増をもたらしたが，処理時間は増加し，ベース手法に方策 2 を単独で施した場合は約 3.4 倍，方策 1 と 3 の組み合わせに方策 2 を追加した場合でも約 1.1 倍の増加が見られた．これはクエリの増加による計算量の増大に依るものである．認識率と処理時間のトレードオフを考慮した上で方策 2 の有無，またはパラメータの設定を変更する必要があることが分かった．方策 3 は認識率の向上と処理時間の削減をもたらした．処理時間については，主にハッシュ探索に要する時間が削減されたと考えられる．また，ベース手法では誤ったビンを参照し，かつそのビンで衝突が多く発生していた場合，大量の誤投票が生じることがあったが，方策 3 により衝突が解消したために誤投票が減り，認識率が向上したものと考えられる．ただし，方策 1 に方策 3 を追加した場合は，既に誤投票が解消されているため，認識率の向上は見られなかった．

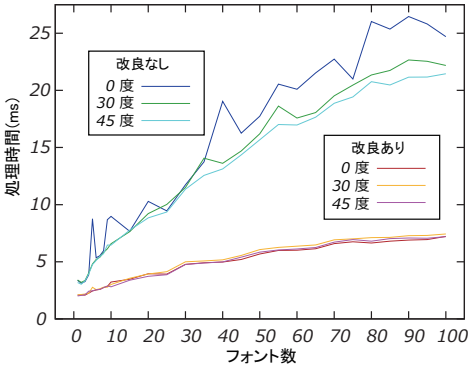
次に，図 15(e) に本実験で用いたデータベースのクラ



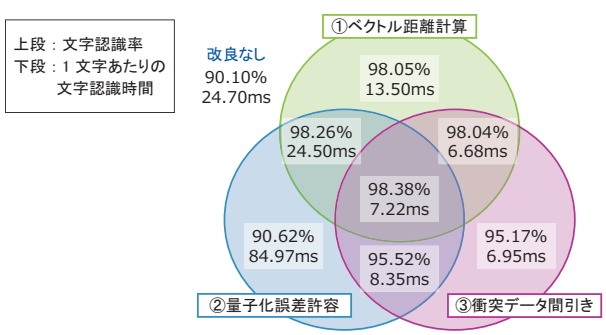
(a) クラス認識率.



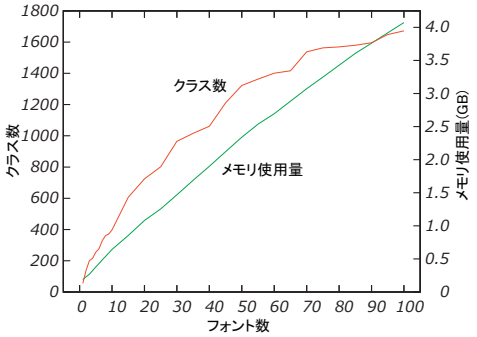
(b) 失敗例.



(c) 1 文字当たりの平均処理時間.



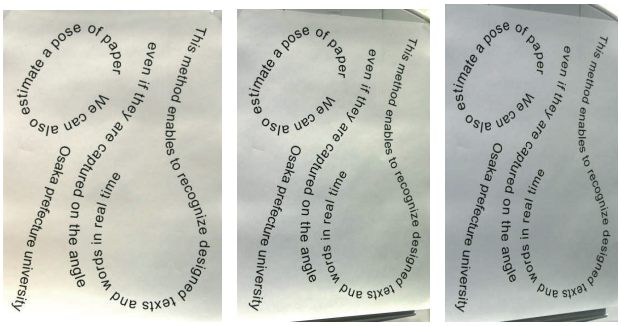
(d) 方策別の文字認識性能（100 フォント登録，正面から撮影時）.



(e) クラス数とメモリ使用量.

図 15: 文字認識結果.

ス数とメモリ使用量を示す．クラス数は，1 フォント登録時に 55 クラス，10 フォントで 397 クラス，100 フォントで 1672 クラスであった．登録フォント数の増加によってクラス数が増加したものの，増加率は徐々に減少した．これは，データベースに存在する画像の増加により，新規登録される画像が既存のクラスに属しやすくなるためと考えられる．一方，メモリ使用量は登録フォント数にほぼ比例して増加した．この原因として，クラスが統合されることとは無関係にハッシュ表に登録される情報が増加するためである．100 フォントの場合のメモリ使用量は約 4GB であったが，実装の工夫によって減少可能と考えられるため，今後の課題とする．



(a) 0 度.　(b) 30 度.　(c) 45 度.

図 16: 単語認識の対象（Arial）

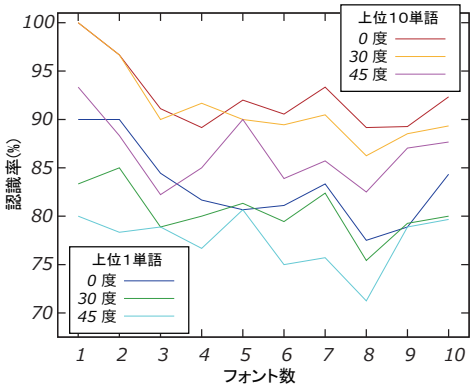
5.2　単語認識に関する実験

単語認識の対象として，図 16 のような画像を用いた．これらは文字認識の対象と同様に文字が曲線上に並ぶようにレイアウトされた文書であるが，内容が英文 30 単語（合計 144 文字）であることが文字認識の対象とは異なる (注4)．その他の条件は文字認識の対象と同じである．図 16(a) の画像の大きさは 1633 × 2333 画素であった．

実験では，データベースと認識対象のフォント数を 1 から 10 まで 1 フォントずつ増やし，単語認識性能の変化を調べた．提案手法を用いて単語の候補を挙げ，ペナルティを昇順に並べた．そして，各単語について最小ペナルティの単語 1 個のみ，または上位 10 個までの単語を見たときに正解の単語が含まれるかどうかを調べ，単語認識率を計算した．なお，先述の通り，本手法では拡大縮小の関係にある字種の区別が困難なため，単語中の”C”，”O”，”S”，”V”，”W”，”X”，”Z” については大文字と小文字の違いがあっても正解とした．

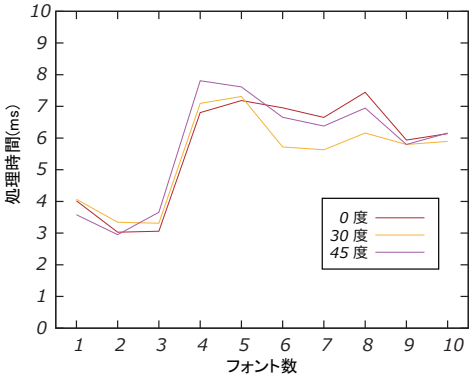
結果を図 17 に示す．まず，図 17(a) は単語認識率である．概ね扱うフォント数の増加に伴い認識率が低下し，1 位（最小ペナルティ）の単語のみを見た場合，10 フォント，0 度では 84.3% であった．10 位までの単語を調べると，角度にも依るが認識率が 10%前後改善し，10 フォント，0 度では 92.3%となった．1 位の単語のみで正解の単語がカバーできない原因は，同クラスで向きが類似した不正解文字が正解文字よりも小さいペナルティを得た

(注4)：文書に 1 つ含まれる冠詞の「a」は単語認識処理を行う必要が無いため，30 単語に含まれていない．



(a) 単語認識率.

(b) 失敗例.



(c) 1 単語当たりの単語認識時間.

図 17: 単語認識結果.

ためと考えられる．なお，20 位までを調べた場合は 10 位の場合より平均 0.4%の上昇に留まったため，これ以上はペナルティの逆転以外の要因による認識失敗である．

単語認識の失敗原因として第一に考えられるのが，文字認識部で誤ったクラスを得てしまったことである．ある 1 文字のクラス識別に失敗すると，他の文字の識別が成功していても 1 単語の認識が不可能になる．そのため，単語認識の精度には 1 文字単位での認識精度が非常に大きく関わると言える．他の原因としては，図 17(b) のように単語領域の取得に失敗したことが挙げられる．文字列を囲む枠線は推定した単語領域の輪郭を表す.「estimate」の「e」とそれ以外の文字が過分割されたことが分かる．この単語を認識できるように画像のぼかしを強くすると，別の文書画像においてスペースを隔てて複数の単語が結合してしまうことを確認している．そのため，ぼかし強度の決定法を変えること，もしくは画像をぼかす方法以外で単語領域を取得することが課題となる．

次に，図 17(c) に 1 単語当たりの処理時間を示す．処理時間は提案手法の単語認識部に要した時間であり，文字認識部は含まない．概ね登録フォント数の増加によって処理時間が増加し，10 フォント，0 度の場合，処理時間は 6.14 ミリ秒であった．処理時間増加の要因としては，クラス当たりの所属画像数が増え，単語推定に要する計算量が増加したためと考えられる．

5.3 実用性に関する性能評価

最後に，実用性の観点から提案手法の性能を評価する．表 2 は，本稿で行った実験結果より提案手法の性能をまとめたものである．フォント数が 10，カメラ撮影角が 0 度とする．5 文字から成る単語を想定すると，文字認識 5 回分と単語認識 1 回分の合計処理時間は 23.34ms であり，1 秒間に約 42 単語の処理が可能である．ただし，認識精度とメモリ使用量については，それぞれの実験の部分で述べた通り，改良の余地を残している．

表 2: 提案手法の性能.（10 フォント登録，正面から撮影時）

単語認識率 (上位 10 単語)	92.3 %
1 単語当たりの単語認識時間	6.14 ms
1 文字当たりの文字認識時間	3.44 ms
メモリ使用量	397 MB

6. むすび

文字認識と単語認識の二段階処理に基づくカメラで撮影した単語画像の実時間認識手法を提案した．文字認識処理においては，クラスレベルの認識ではあるものの，認識対象の文字画像に劣化（射影歪みや解像度の低下，ぼけ）が生じるという条件の下で認識率と処理時間の両方において高いレベルで達成できる手法を実現した．単語認識処理においては，単語辞書を使用せず，推定した文字の方向のみから単語を推定する手法を提案した．日本語への対応，単語辞書の利用等が今後の課題である．

謝辞 本研究の一部は，科研費補助金若手研究 (B) 21700202 ならびに平成 21 年度シーズ発掘試験（発掘型）（課題番号 11-084）の補助による．

文　献

[1] G.K. Myers, R.C. Bolles, Q.-T. Luong, J.A. Herson, and H.B. Aradhye, “Rectification and recognition of text in 3-d scenes,” IJDAR, vol.7, no.2-3, pp.147–158, 2004.

[2] 岩村雅一，辻　智彦，堀松　晃，黄瀬浩一，“レイアウト非依存な実時間カメラベース文字認識,” 画像の認識・理解シンポジウム (MIRU2009) 論文集，pp.174–181, July 2009.

[3] 野口和人，黄瀬浩一，岩村雅一，“大規模特定物体認識における認識率，処理時間，メモリ量のバランスに関する実験的検討,” 電子情報通信学会論文誌 D，vol.J92-D, pp.1135–1143，Aug. 2009.

[4] Y. Lamdan and H.J. Wolfson, “Geometric hashing: a general and efficient model-based recognition scheme,” Proc. ICCV1988, pp.238–249, 1988.

[5] H.J. Wolfson and I. Rigoutsos, “Geometric hashing: an overview,” IEEE Computational Science and Engineering, vol.4, no.4, pp.10–21, 1997.

[6] H. Ishida, S. Yanadume, T. Takahashi, I. Ide, Y. Mekada, and H. Murase, “Recognition of low-resolution characters by a generative learning method,” Proc. CBDAR2005, pp.45–51, 2005.