カメラで撮像した文字画像の実時間認識システム

A Real-Time Recognition System of Camera-Captured Characters

岩村 雅一† 辻 智彦† 堀松 晃† 黄瀬 浩一†
Masakazu Iwamura†, Tomohiko Tsuji†, Akira Horimatsu†, Koichi Kise†
† 大阪府立大学大学院工学研究科
†Graduate School of Engineering, Osaka Prefecture University
E-mail: masa@cs.osakafu-u.ac.jp

Abstract

本稿ではカメラを用いた文字認識技術の応用を見据えて，web カメラと接続したノートパソコン上で実時間で動作することができるカメラベース文字認識技術を提案する．提案手法は，実時間処理が可能であること，射影歪みに対応すること，様々なレイアウトの文書が認識可能であるという 3 つのを同時満たす実用的な技術である．

1 Introduction

カメラを用いた文字認識は様々な応用を見込めることから，近年注目を集めている．有望な応用の一つとしては，カメラと文字認識装置を組み合わせた翻訳装置である「翻訳カメラ」がある [1]．また，カメラで写した文字を認識し，音声に変換することで視覚障害者に伝える応用も可能である．さらにカメラに写る全ての文字を認識し，その中で事前に登録しておいた利用者が必要とする情報のみを利用者に伝える応用も考えられる．この応用は視覚障害者にとって有益であるといえる．このような応用を実用化するためには，実時間処理が可能であること，射影歪みに対応すること，様々なレイアウトの文書が認識可能であるという要件を満たす実用的な文字認識技術が必要である．

まず実時間性は利用者の利便性を低下させないために絶対必要である．射影歪みへの対処については，文献 [2, 3] の手法が実現していて，特に文献 [3] の技術は実時間で動作することが報告されている．これらの手法では，まずカメラを用いて撮像した画像から文字行を抽出し，次に射影歪みや射影歪みの近似であるアフィン歪みを補正し，最後に切り出された文字を認識する技術が提案されている．しかし，これらの技術は文字認識の前処理段階で一旦文字行を抽出するため，文字行を成していない文字は認識できない．そのため，図 1 のような文字行が直線でない場合も認識することができない．

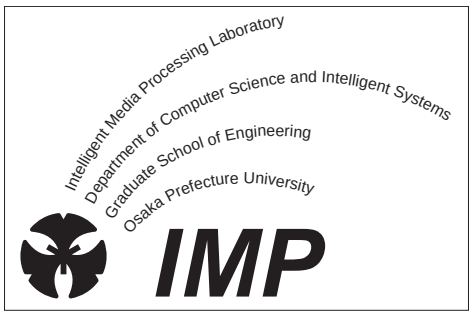


図 **1** 文字行が直線でないようにデザインされた文書の例．

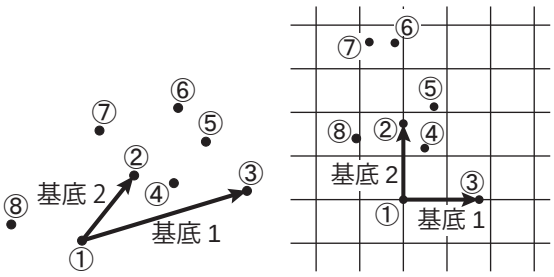
一方，文献 [4, 5] の手法は文字を 1 文字ずつ認識するため，前述の文字行の問題は起こらないが，処理に時間がかかり，実時間で動作しない．

そこで本稿では，前述の 3 要件を満たした文字認識技術を実現するために，実時間で文字を認識することができる単純だが効果的な手法を提案する．提案手法の文字切り出しには適応二値化を施し，輪郭抽出を行う．提案手法の文字認識には，Geometric Hashing(以下，GH) を改良して用いる．アフィン変換を考慮した GH の計算量は P を点数としたとき，$O(P^4)$ であるが，提案手法では不変量の計算原理を利用して $O(P^2)$ にまで削減可能である．さらに投票機構を利用した手法を組み合わせて利用することで，提案手法は web カメラと接続したノートパソコン上で実時間で動作することができる．

2 Geometric Hashing の改良

2.1 Geometric Hashing [6]

GH は幾何歪みを受けた画像を不変座標系を用いて記述し，認識する強力な手法である．本稿で提案する GH の改良手法を説明するために，アフィン変換を考慮



(a) 基底の決定. (b) 不変座標系への射影.

図 **2** **Geometric Hashing** の不変座標系**.**

した GH を簡単に説明する.

2.1.1 登録処理

登録する参照画像から抽出した特徴点が与えられているとする．まず，全特徴点から 3 点選び，図 2(a) に示すように選ばれた特徴点の順番を考慮して 2 本の基底ベクトルを作成する．そして，2 本の基底ベクトルを用いて図 2(b) のように新しい座標系を作成し，特徴点を写像する．この座標系は画像がアフィン変換を受けても同様に作成できるため，アフィン不変座標系である．このアフィン不変座標系を図 2(b) のように格子状に区切ると，各領域は 2 次元ハッシュテーブルのビンに相当する．各特徴点が存在するビンに対して，登録する画像の番号と基底の組の通し番号を登録する．この処理を全ての可能な基底に対して実行し，1 枚の参照画像の登録が終了する．全ての参照画像を登録して登録処理が終了する.

アフィン不変座標系の作成に $O(P^3)$，特徴点の射影等に $O(P)$ の計算量が必要であるため，参照画像 1 枚当たりの計算量は $O(P^4)$ である.

2.1.2 認識処理

認識対象である質問画像から抽出した特徴点が与えられているとする．認識処理の前半は登録処理と同じである．全特徴点から 3 点選び，図 2(a) に示すように，選ばれた特徴点の順番を考慮して 2 本の基底ベクトルを作成する．そして，2 本の基底ベクトルを用いてアフィン不変座標系を作成する．このアフィン不変座標系は登録時に格子状に区切られていて，各領域が 2 次元ハッシュテーブルのビンに相当する．各特徴点が存在するビンから，登録されている画像の番号と基底の組の通し番号を取り出して，画像の番号と基底の組の通し番号に対して投票する．この処理を全ての可能な基底に対して実行し，最大の投票数を得た画像の番号と基底の組の通し番号を決める．そして，この画像の番号を認識結果として出力する．ただし，全ての基底を処理する前に結果が明らかになったときには処理を途中で終了することができる.

アフィン不変座標系の作成に $O(P^3)$，特徴点の射影等に $O(P)$ の計算量が必要であるため，合計の計算量は $O(P^4)$ である.

2.2 提案する Geometric Hashing の改良手法

2.2.1 問題設定の違い

提案手法は GH の改良手法である．提案手法について述べる前に，GH と提案手法の問題設定の違いについて述べておく．GH では，特徴点が与えられたときに点の配置のみから対象を同定する問題を解いている．つまり，特徴点がどのような対象からどのように抽出されたかを考慮しない．それに対して提案手法では，図形が与えられたときに図形から得られる特徴点の配置と，図形の特徴の両方を用いて図形を同定する．そのため，提案手法では原則として図形の輪郭上の画素全てを特徴点とする.

2.2.2 計算量の削減

GH の欠点は膨大な計算量である．アフィン変換に対応した場合，認識処理に必要な計算量は点数 P に対して $O(P^4)$ である．$P = 100$ 点の場合を考えると $O(100,000,000)$ もの計算が必要になるため，実時間アプリケーションに使用することは現実的でない．一方，提案する方法を用いれば，最も計算量が少ない場合，$O(P^2)$ にまで削減することができる.

GH の計算量が膨大である理由を述べる [7]．GH では認識が成功するために，登録処理で用いた基底の組が認識処理でも計算される必要がある．しかし，基底の組が一致するかどうかは計算してみるまでわからないため，結局全て (もしくは多数) の基底を計算することになってしまう．したがって，もし登録処理と認識処理で同一の基底を計算することができれば計算量を削減することができる．そこで提案する GH の改良手法では，登録処理と認識処理で同一の特徴点を選択することにより，同一の基底を計算するために必要な計算量を削減している.

提案手法で計算量を削減する特徴点の選択方法について述べる．アフィン変換の場合は図形の重心が保存されるため，重心を 1 点目の特徴点とする[1]．2 点目は GH と同様，適当に選択する.

3 点目は，ここまでに得られた 2 点とこれから述べる不変量が持つ性質を用いて自動的に定める．最初に，最も単純な図 3 の例を用いて不変量が持つ性質につい

[1]重心が輪郭上になる保証はないが，特徴点が輪郭以外に存在しても後の処理に影響はない.

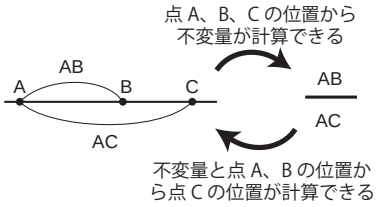


図 **3**　点の位置とアフィン不変量の関係.

て説明する．図 3 のように 1 直線上に 3 点 A，B，C が与えられたとするとき，AB/AC はアフィン変換を受けても変化しない不変量である．このように点の座標から不変量の値を計算することが通常行われる．これに対して提案手法では，不変量の値と 2 点 A，B が与えられたときに点 C の座標を求める．点 C が点 A，B と同一直線上にあるとすると，点 C の位置は点 A の左側にある場合と，図 3 のように点 B の右側にある場合の両方が考えられるが,「直線上に A，B，C の順番に並ぶように点 C を定める」といった具合に点 C の決定方法を事前に定めておけば一意に定めることが可能になる．この原理を一般化して記述すると,「n 点の座標から計算した不変量の値と $n-1$ 点の座標が与えられれば，残る 1 点 (n 点目) を計算することができる」となる．

上記のように基底の作成に用いる特徴点を一意に決定できれば計算量を削減できる．上記の方法は 2 点を一意に決定したので，計算量を $O(P^4)$ から $O(P^2)$ に削減することができる．

なお，上記の特徴点決定方法を用いて 3 点目を定めたとしても 3 点が一直線上に存在してしまうため，2 本の一次独立な基底を作成することができない．以下では，図 4(a) のような面積が S_0 である図形の 3 点目を決定できる上記とは別の方法を述べる．

1. 方法 1
   図 4(b) のように 3 点の特徴点が与えられたとする．1 点目と 2 点目を通る半直線と，1 点目と 3 点目を通る半直線を考え，図形から切り取られる面積を S_1 とする．このとき，S_1/S_0 がアフィン不変量になるので，S_1/S_0 が特定の値になるように 3 点目を定めればよい.
2. 方法 2
   方法 1 と同様に，図 4(c) のように 3 点の特徴点が与えられたとする．3 点が作る三角形の面積を S_1 としたとき，S_1/S_0 がアフィン不変量になる．したがって，S_1/S_0 が特定の値になるように 3 点目を定めればよい．S_1/S_0 の値は，特定の値でなくとも，最大値でもよい．なお，S_1 が一定になるように 3 点目を定めることを考えると，取り得る 3 点目の軌跡は図 4(c) のように 1 点目と 2 点目を通る直線と平行な直線になる．したがって，この直線と図形との交点を 3 点目に定めればよく，簡便に計算可能である．交点が複数ある場合は 2 点目から近いほうを選ぶといった選択方法も可能である.

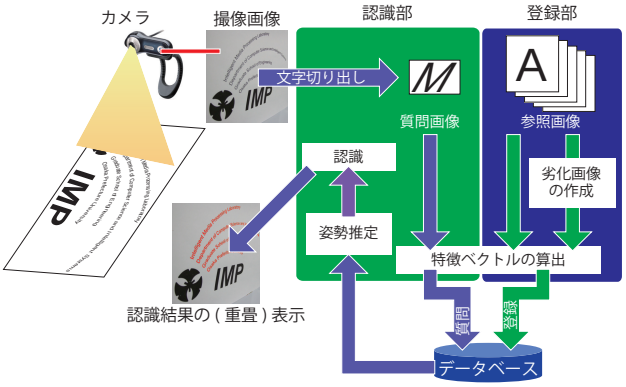


図 **4**　提案システムの概要.

ところで，上記の方法とは別の方法で最初の 2 点を決めることもできる (方法 3)．すなわち，1 点目を GH と同様に，P 点から適当に選択し，2 点目を決める際に面積比を利用して決めることもできる．図 4(d) のように 2 点の特徴点が与えられたとすると，面積比 S_1/S_0 がアフィン不変量になる．したがって，S_1/S_0 が特定の値になるように 2 点目を定めればよい.

本稿の実験では，方法 2 で最大値を用いる方法を採用する.

2.2.3　図形の特徴量の使用

GH は画像番号と基底番号の組をデータベースに登録する．一方，提案手法では基底番号の代わりに，画像から計算した特徴ベクトルと基底の作成に用いた特徴点の座標を登録する.

画像から計算した特徴ベクトルを用いるのは，画像の特徴のほうが表現力が高いと考えられるからである．GH の問題設定では認識対象から抽出された特徴点の座標のみが与えられたが，提案手法が考える問題では認識対象の図形そのものが与えられている．そのため，図形から抽出した特徴ベクトルが利用可能になった.

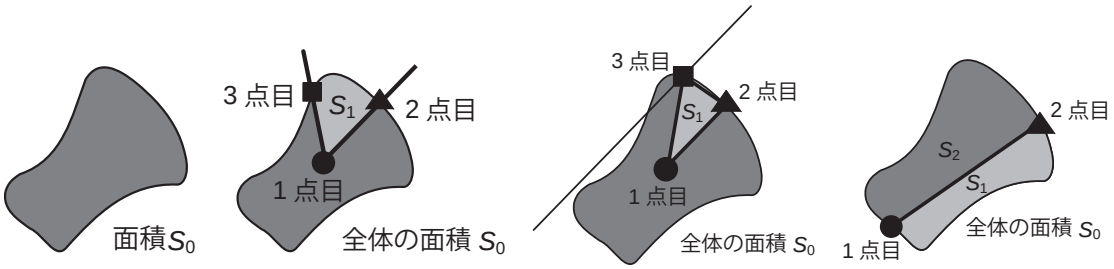
以後は文字認識の話に特化するため,「画像番号」のことを「文字番号」と呼ぶことにする.

3　提案システムの概要

提案システムの概要を図 4 に示す．提案システムは画像登録部と画像認識部から成る.

3.1　画像登録部

画像登録部では，参照画像をデータベースに登録する．参照画像は二値画像とする.



(a) 元の図形. (b) 方法 1. (c) 方法 2. (d) 方法 3.

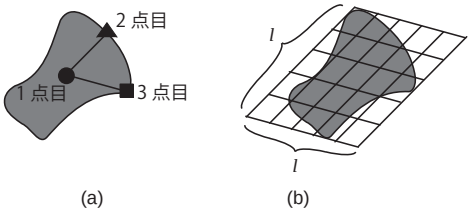


図 **5**　特徴ベクトルの計算方法. **(a) 3** 本の直線から **2** 本の直線を選択すると，**(b)** $k = l \times l$ としたとき，k 個の部分領域のヒストグラムを利用して k 次元の特徴ベクトルを作成できる.

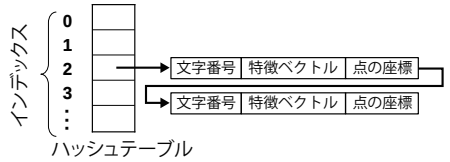


図 **6**　ハッシュ表の構成.

3.1.1　劣化画像の生成

撮影時のピンぼけに対処するため，参照画像にガウスぼかしを重畳する生成型学習法 [8] を適用した．本稿では，畳み込む正規分布の標準偏差 σ を $\sigma = 2, 4, 6$ と変えながら，参照画像 1 枚につき，ぼけの程度が異なる 3 枚の劣化画像を生成した．生成された劣化画像は再び二値化し，以後は参照画像と同様に扱う.

3.1.2　特徴ベクトルの生成

2.2.2 で述べた方法で 3 点の特徴点を選択し，不変座標系を生成することにより，特徴ベクトルを生成する.

特徴点が 2 点あれば，2 点を通る直線を 1 本定めることができるので，3 点から合計 $\binom{3}{2} = 3$ 本の直線が計算できる．3 本の直線のうち 2 本を選ぶことにより，図 5 に示すような k 個の均一な部分領域を設定できる．ここで $k = l \times l$ である．このように部分領域を設定することは，図 2(b) のように GH で 2 本の基底で定められる不変座標系において格子を設定することと同じである．各領域内の特徴点数を数え上げ，合計が 1 になるように正規化することで k 次元特徴ベクトルが計算できる．本稿では，各領域の値の計算に，輪郭上の画素のみでなく，図形内部の全画素を用いた．3 本の直線から順番を考慮して 2 本選ぶ組み合わせは 3 通りなので，3 本の k 次元特徴ベクトルが計算できる．それらを単純に結合することで $3k$ 次元の特徴ベクトルが得られる.

3.1.3　データベースへの登録

データベースは，具体的にはハッシュテーブルで構成されている.

データベースへの登録方法について述べる．文字番号，特徴ベクトル，3 点の特徴点の座標を組にして，ハッシュテーブルに格納する．提案手法では特徴ベクトルを使用するため，ハッシュテーブルは GH のような 2 次元ではなく，1 次元になる．ハッシュのインデックス H_{index} は次式で計算する.

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

ここで H_{size} はハッシュテーブルの大きさ，r_i は特徴ベクトルの i 番目の要素を $D = 2$ 通りに離散化した値である．なお，衝突が起こる場合は図 6 に示すようにリスト構造で連結する.

3.2　画像認識部

3.2.1　画像の取得

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び，以下の処理で用いる.

3.2.2　文字画像の切り出し

得られた画像から文字画像を切り出す．まず画像に対して，適応二値化を施す．すなわち，注目画素を中心とする一辺が n の矩形の平均輝度より明るいか暗いかによって白 (輝度 1) または黒 (輝度 0) を決定する．本稿では，$n = 101$ とした.

次に，連結成分[2]を抽出する．得られた連結成分を文字領域候補とみなして，矩形で切り出し，以下で述べる認識処理の対象とする．ただし，得られた連結成分の面積が閾値以下であればノイズとみなして認識対象から除外する．

3.2.3 特徴ベクトルの生成

得られた連結成分から特徴ベクトルを生成する．この処理は 3.1.2 で述べた処理と基本的に同じである．唯一異なるのは，全ての可能な組み合わせに対して不変座標系を生成せず，あらかじめ定める S 個に限定することである．

3.2.4 投票を用いたパラメータ推定と認識

投票を用いてパラメータ推定と認識を行う．最初に，文字番号，特徴ベクトル，3 点の特徴点の座標の組をハッシュテーブルから S 個得る．そして，文字番号 i に対して重み $\frac{1}{\sqrt{P_i}}$ を用いる重み付き投票を行う．重みを用いる理由は，文字毎に特徴点数 (輪郭の長さ)P_i が異なり，特徴点が多い文字は得票が不公平に多くなると考えられるからである．重み付き投票によって得られた最大の得票数を M_i とおく．この値を基に，重み付き投票から 2 つのグループを定義する．1 つ目は得票数が $0.9M_i$ 以上の文字のグループで,「推定グループ」と呼ぶ．2 つ目は得票数が $0.8M_i$ 以上の文字のグループで,「候補グループ」と呼ぶ．

質問画像から得られた 3 点の座標とデータベース中の 3 点の座標の対応関係から 1 つのアフィン変換行列が求まる．質問画像から得られた 3 点の座標は S 組あるため，合計 S 個のアフィン変換行列が得られる．各アフィン変換行列を文献 [9] の要領で拡大率 β，回転角 θ，せん断変形の度合 φ，独立変倍の倍率 α を 4 つのパラメータに分解する．

本稿では全ての文字は同一平面上に存在すると仮定している．この場合，せん断変形の度合 φ と独立変倍の倍率 α は全ての文字で共通になるはずである．そこで，φ と α が作る 2 次元空間での密度推定を利用して，尤もらしい φ と α の組を推定する．ここでは,「推定グループ」に属する文字のアフィン変換行列を前述の 2 次元空間にプロットする．プロットされた点の中から近傍の密度が最も高い点を選択する．この処理は，まず φ と α のそれぞれを 5 等分し，2 次元空間を 25 等分する．25 個の部分領域について，その領域と 8 近傍の合計 9 領域に含まれる点の数を集計し，点の数をその領域のスコアとする．全部分領域についてスコアを計算した後，最もスコアの高い領域を選択する．この領域に含まれる点の数が 30 個より多ければ，その領域をもう一度 25 等分し，同様の処理を点数が 30 個以下になるまで繰り返す．スコアが最大の領域に含まれる特徴点数が 30 個以下になったとき，その領域の中心の値を φ と α の推定値とし，$\hat{\varphi}$ と $\hat{\alpha}$ とおく．

最後に，連結成分ごとに認識結果を定める．前述の φ と α が作る 2 次元空間において,「候補グループ」に属する文字のアフィン変換行列の中で $(\hat{\varphi}, \hat{\alpha})$ から最も近い点を選び，そのアフィン変換行列を与えた文字を認識結果 (第 1 位候補) とする．もし認識結果が 2 つ必要な場合は，第 1 位候補を除いて $(\hat{\varphi}, \hat{\alpha})$ から最も近い点を選び，第 2 位候補とする．以下，同様の処理を繰り返す．

4 実験

4.1 各種フォント

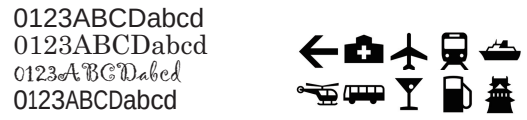
提案手法の有効性を調べるために様々なフォントの文字を認識した．字種には 60 種類の数字とアルファベットを用いた．内訳は，10 種の数字，“i” と “j” を除いた 24 種の小文字アルファベット，26 種の大文字アルファベットである．一部の文字はアフィン歪みを受けると外見が類似してしまうため，表 1 で同じ箱に入っている文字は認識実験において同一クラスとみなした．例えば，0(ゼロ) を O(オー) に間違えても誤認識とはしない．

実験には図 7(a) に示す Arial，Century，Gigi，Impact の 4 種類のフォントを用いた．認識対象として，図 8 に示すテストパターンを作成した．このテストパターンには文字の大きさが 3 種類 (72pt，48pt，32pt)，文字の傾きが 3 種類 (0 度，30 度，45 度) の 9 種類の条件が含まれている．1 条件につき 12 文字ずつ含まれているので，合計 108 文字が含まれている．これを 60 字種分，60 枚作成した．印刷したテストパターンは，デジタルカメラの紙面に対する傾きが 0 度，30 度，45 度の 3 種類になるように撮影した．撮影した解像度は 1024×768 である．実験のパラメータとしては，$S = 200$，$k = 25$ を用いた．

計算量を削減するために，画像登録時には参照画像から連結成分を抽出し，幅と高さの大きいほうが 100 ピクセルになるように大きさを正規化した．画像認識時には，質問画像から抽出した各連結成分の幅と高さの大きいほうが 50 ピクセルになるように大きさを正規化した．実験には，CPU が Opteron 2.4GHz で，メモリが 128GB である計算機を用いた．

累積認識率と平均処理時間を図 9 と表 2 に示す．提案手法はアフィン不変であるが，カメラを傾けた認識実験の結果から射影変換に対してもある程度頑健に認識できていることがわかる．Arial，Century，Gigi の各フォントは順位の増加と共に累積認識率が増加して

[2]画像中で黒画素が隣接し，一塊になっているものを指す．



(a) 上から Arial，Century，Gigi，Impact.　(b) 10 種のピクトグラム.

図 **7**　実験に用いたフォントとピクトグラム.

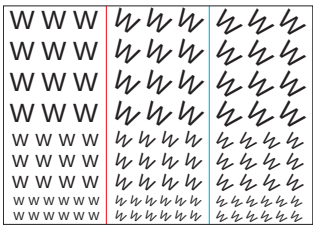


図 **8**　認識対象の紙面の例.

いるが，おおよそ 6 位程度で認識率が飽和している．一方，Impact は 1 位認識率が非常に悪いが，20 位まで累積認識率が上昇し続けている.

4.2　ピクトグラム

図 7(b) に示す 10 種のピクトグラムを 4.1 と同様に認識実験を行った.

認識率と処理時間を図 10 に示す．ビン数が 16 のときに認識率が最高になった．処理時間はほとんどの場合で同程度だったが，ビン数が 4 のときに極めて大きくなった．それと同時に認識率が最低になった．これは特徴ベクトルの識別能力が十分でないため，ハッシュに多数の衝突が発生したためと思われる.

4.3　図 1 の文書

最後に，図 1 に示した文書を認識した．デジタルカメラを紙面から 0 度，30 度，45 度の 3 種類に傾けて撮影して，背景が写らないように紙面の部分だけ切り取った．切り取った後の 0 度，30 度，45 度の画像の大きさはそれぞれ 2054×1464，1714×1326，1516×1322 である．得られた画像を図 11 に示す．図 1 に含まれる文字から得られる連結成分は 148 個であるが，そのうち 18 個は “i” と “j” の一部であった．“i” と “j” は連結成分が 1 つではないため，参照画像に含まれていない．し

表 **1**　アフィン歪みを受けた場合の類似文字のリスト．同じ箱に入っている文字はアフィン歪みを受けた場合に認識が難しいので，同一のクラスとした.

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

表 **2**　1 文字の認識に要する平均処理時間

フォント	Arial	Century	Gigi	Impact
平均処理時間 (ms)	32.4	24.7	24.0	81.0

(a) 0 度　(b) 30 度　(c) 45 度

図 **11**　図 **1** の画像をデジタルカメラで撮像して得た認識対象.

たがって，これらの 2 字種を認識することができない．そこで，残る $148 - 18 = 130$ 個の文字の認識率を算出した．それ以外の条件は 4.1 と同様である.

認識率と処理時間を表 3 に示す．$S = 200$ の場合は $S = 20$ の場合よりも認識率が高かった．$S = 20$ の場合の処理時間は $S = 200$ の場合の約 1/8〜1/7 であったが，認識率の差はそれほど大きくなかった．$S = 20$ の場合の結果から，提案手法が高速で頑健な認識が可能であることを確認できた.

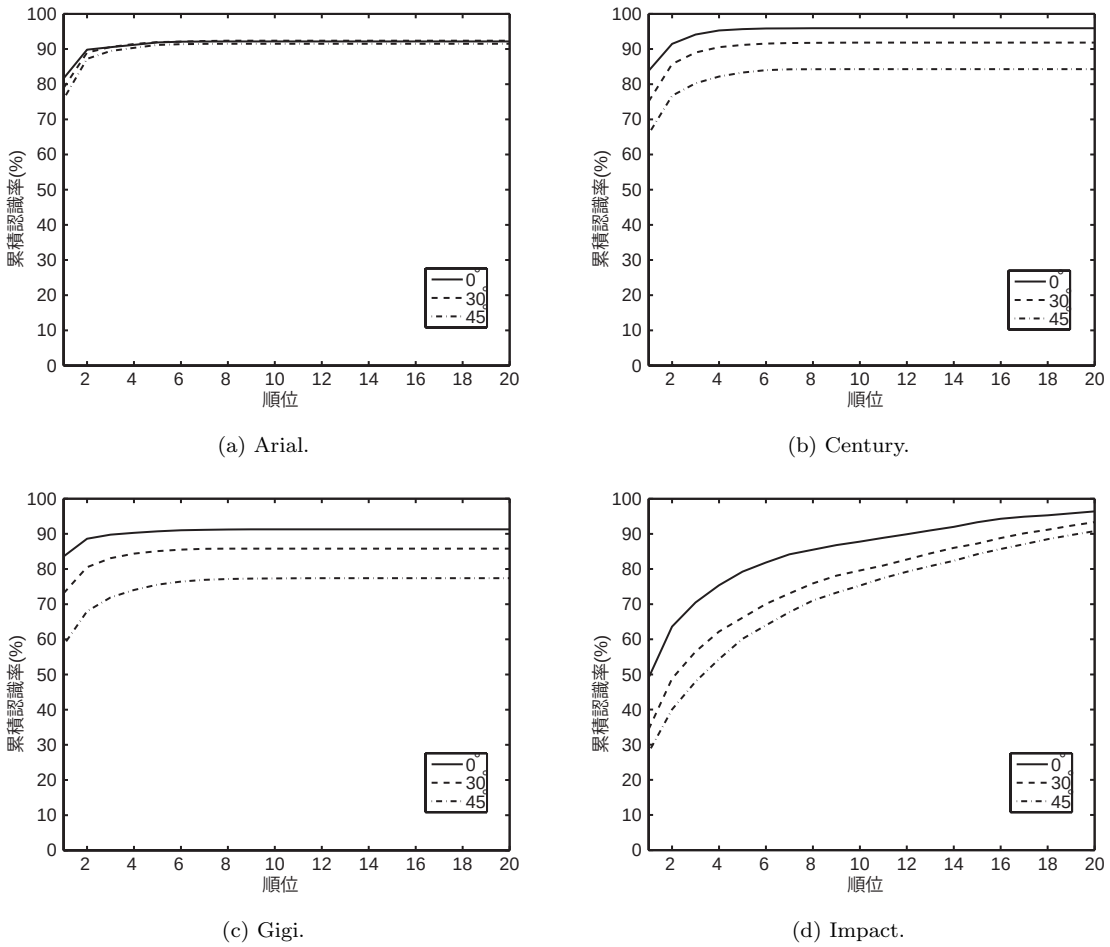
5　関連研究

文字認識手法以外の関連研究をまとめておく.

中居らが提案した Locally Likely Arrangement Hashing(LLAH) という手法がある [10]．この手法は，特徴点の局所的な配置に着目し，幾何学的不変量とハッシュを用いてデータベース中から対応する特徴点を高速に検索する手法である．GH と LLAH を比較すると，LLAH は検索の頑健性を保ちつつ，計算量とメモリ使用量を従来手法の数億分の一に減少させている．この性能向上を可能にしているのは，特徴点の選択方法を限定して計算量を減少させていることと，特徴ベクトルの高次元化による識別性能の向上である．前者は提案手法の考え方と類似しているが，LLAH では特徴点が分散していることを想定しているため，提案手法のように特徴点が連続している場合には適用できない．後者については，提案手法にも適用することができ，提案手法の更なる性能向上が見込める．その方法を説明する

表 **3**　図 **1** を認識したときの認識率と処理時間

S	200			20		
角度 (度)	0	30	45	0	30	45
認識率 (%)	91.5	93.1	86.9	90.0	86.2	83.9
時間 (ms)	5400	5210	4790	740	710	640



(a) Arial.　(b) Century.

(c) Gigi.　(d) Impact.

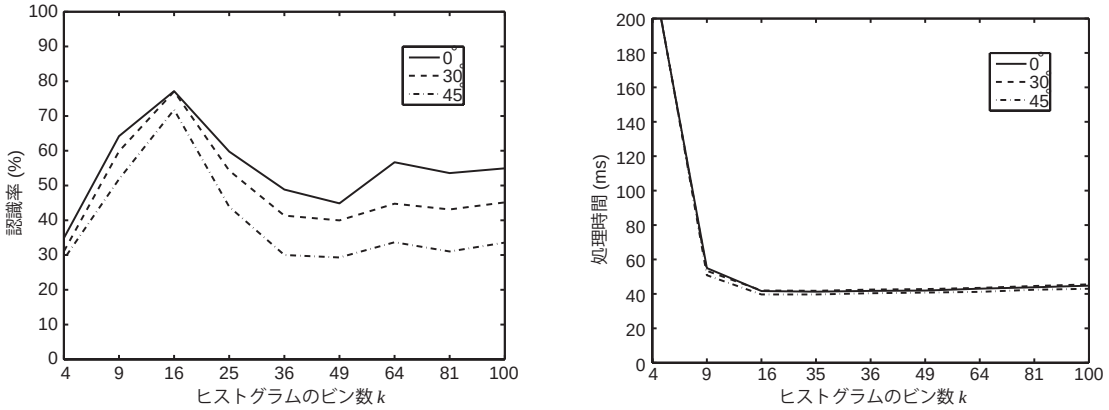
図 **9**　様々なフォントに対する累積認識率

ために，まず LLAH がどのように特徴ベクトルを高次元化しているかを説明する．アフィン変換の場合，同一平面上に 4 点あれば 1 つの不変量が計算できる．LLAH では，$m > 4$ 点から 4 点を選ぶ $\binom{m}{4}$ 個の不変量を計算して，$\binom{m}{4}$ 次元ベクトルを作成することで特徴ベクトルを高次元化し，識別性能を向上させている．提案手法でも LLAH と同様の処理が可能である．すなわち，3 点より多数の特徴点を求めて，多数の特徴ベクトルを計算する．そして，それらを全て結合して高次元化する．これにより，より識別性能の高い特徴ベクトルを計算することができる．

提案手法や LLAH と同様に，特徴の選択方法を限定して計算量を減少させている方法が他にもある．Rothwell らによって提案された手法 [11] では，連結した 5 本の線分や 2 個の円錐曲線を抽出することで，射影変換の不変量を計算している．抽出された線分などは隣接しているため，簡単に順序付け可能であり，不変量を計算する順番の組み合わせが限定されるため，高速化が可能である．しかし，本稿で対象とする任意形状の図形から常に線分やコニックを抽出できるとは限らないため，この方法は適用が困難である．

また，アフィン変換を受けた図形そのものを正規化し，その後に照合する方法もある．Leu らは図形を構成する画素の座標値から共分散行列を求め，その逆行列を用いることで拡大・縮小，せん断変形を正規化する方法を提案している [12]．しかし，この方法では回転に任意性を残すため，何らかの方法で図形の回転を考慮した照合が必要である．この問題に対して Horimatsu らは回転方向の照合を高速化する手法を検討しているが，この手法でも十分高速とは言い難い [13]．この問題に対して，提案手法のアフィン変換の方法を用いるか，提案手法を相似変換レベルで用いれば，どちらの場合も $O(P^2)$ の計算量で照合可能である．計算量の内訳は，相似不変座標系の作成が $O(P)$，特徴点の射影等が $O(P)$ である．

相似変換の場合の具体的な方法は，1 点目，2 点目はアフィン変換のときと同様に求める．3 点目については，相似変換では角度が保存されることから，あらかじめ定められた角度を成すように定める方法や，長さが保存されることから，1 点目，2 点目からあらかじめ定められた距離になるように定める方法などが考えられる．



(a) Recognition rates.

(b) Processing time.

図 **10**　特徴ベクトルの大きさを変化させたときのピクトグラムの認識率と処理時間

6　むすび

実時間処理が可能であること，射影歪みに対応すること，様々なレイアウトの文書が認識可能であることを同時に実現するカメラを用いた文字認識を実現するために，本稿では実時間で文字やピクトグラム等を認識することができる単純だが効果的な手法を提案した．本稿では結果を示していないが，提案手法は web カメラと接続したノートパソコン上で実時間で動作することができる．

参考文献

[1] Y. Watanabe, Y. Okada, Y.-B. Kim, and T. Takeda, “Translation camera,” Proc. ICPR1998, pp.613–617, 1998.

[2] X. Chen, J. Yang, and A. Waibel, “Automatic detection and recognitionof signs from natural scenes,” IEEE Trans. Image Processing, vol.13, no.1, pp.87–99, Jan. 2004.

[3] 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.

[4] Y. Kusachi, A. Suzuki, N. Ito, and K. Arakawa, “Kanji recognition in scene images without detection of text fields—robust against variation of viewpoint, contrast, and background texture—,” Proc. ICPR2004, pp.457–460, Sept. 2004.

[5] L. Li and C.L. Tan, “Character recognition under severe perspective distortion,” Proc. ICPR2008, 2008.

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

[7] M. Iwamura, T. Nakai, and K. Kise, “Improvement of retrieval speed and required amount of memory for geometric hashing by combining local invariants,” Proc. BMVC2007, vol.2, pp.1010–1019, Sept. 2007.

[8] 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.

[9] M. Iwamura, R. Niwa, A. Horimatsu, K. Kise, S. Uchida, and S. Omachi, “Layout-free dewarping of planar document images,” Proc. DRR XVI, 7247-36, Jan. 2009.

[10] 中居友弘，黄瀬浩一，岩村雅一，“特徴点の局所的配置に基づくデジタルカメラを用いた高速文書画像検索,” 信学論 D，vol.J89-D，no.9，pp.2045–2054，Sept. 2006.

[11] C.A. Rothwell, A. Zisserman, D.A. Forsyth, and J.L. Mundy, “Using projective invariants for constant time library indexing in model based vision,” Proc. BMVC, 1991.

[12] J.-G. Leu, “Shape normalization through compacting,” Pattern Recognition Letters, vol.10, no.4, pp.243–250, 1989.

[13] A. Horimatsu, R. Niwa, M. Iwamura, K. KIse, S. Uchida, and S. Omachi, “Affine invariant recognition of characters by progressive pruning,” Proc. DAS2008, pp.237–244, Sept. 2008.