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.

Bloomier Filterを用いた特定物体認識手法の性能に関する実験的検討

井上　勝文†　　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:　†inoue@m.cs.osakafu-u.ac.jp, †kise@cs.osakafu-u.ac.jp

あらまし　局所特徴量を表す特徴ベクトルを用いた物体認識手法では，膨大な数の特徴ベクトルを保存するために，莫大なメモリ容量が必要となる．この問題を解決するために，我々は空間効率の良い Bloomier Filter を用いた手法を提案し，メモリ使用量の削減手法としての有効性を示している．しかし，特定物体認識手法の従来法と比べ，どの程度有効性があるのかという疑問が残っており，検討が必要である．そこで本稿では，従来提案されている “bag of features”モデルに基づく手法と比較し，提案手法の有効性について検討する．具体的な結果として，1 万個の平面物体を対象とした実験より，同じ認識率を得るために必要なメモリ使用量を 1/7 にできる．また，提案手法で用いている認識の評価基準と物体認識率の関係を実験的に解析した結果について述べる．

キーワード　Bloomier filter，Bloom filter，メモリ削減，特定物体認識，局所特徴量

Experimental Investigation of the Performance of a Specific Object Recognition Method with the Bloomier Filter

Katsufumi INOUE† and Koichi KISE†

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

Abstract　Specific object recognition based on local feature vectors requires a huge amount of memory to store all feature vectors. To solve this problem, we propose a memory reduction method with the Bloomier filter and show the its efficiency method for memory reduction in the previous paper. However, it is still unclear how efficient the proposed method is in comparison with conventional methods. In this report, we compare the efficiency of the proposed method with that of the method which is based on “bag-of-features”model. From the experiment with 10,000 planar objects, the proposed method allows us 1/7 required memory to obtain the same recognition rate compared with the conventional method. Additionally, in this report, we analyze experimentally the relationship between the recognition criteria with the proposed method and recognition accuracy.

Key words　Bloomier filter, Bloom filter, Memory reduction, Specific object recognition, Local feature

1. はじめに

特定物体認識とは，物体のインスタンスを認識するタスクである．本稿では，PCA-SIFT [1] などの局所特徴量を用いた特定物体認識について述べる．

我々の周囲は様々な物体で溢れているため，特定物体認識を真に実用的なものにするには，大量の物体を高精度で認識することが必須である．画像の局所領域から得られる局所特徴量は，高い識別性を有しているため，大規模物体認識に適している．物体認識を実現するための基本的な考え方は，未知の物体から得られる局所特徴量と，既知の物体から得られる局所特徴量の照合である．一般に，局所特徴量は多次元の実数値ベクトルで表され，照合には特徴ベクトルの最近傍探索が用いられる．この枠組みを用いた最も単純な物体認識手法に，投票処理に基づくもの [2] がある．この手法は，単純であるが高い認識率を得ることができる．しかし，大規模な物体認識を行うためには，乗り越えなければならない課題がいくつかある．最も代表的なものは，処理時間とメモリ使用量に関するものであろう．

前者の問題では，特徴ベクトルの最近傍探索の効率化が必須となる．幸い，特徴ベクトルの近似最近傍探索を導入することにより，処理時間を大幅に改善できることが知られている [2]．

一方，メモリ使用量に関する問題は簡単ではない．一般に，

1 枚の画像から数百から数千個の特徴ベクトルが得られるため，大規模な物体認識を行うためには，膨大なメモリが必要となる．これまでに，特徴ベクトルのベクトル量子化 [3] やスカラ量子化 [2] 等のメモリ削減手法が提案されている．しかし，特徴ベクトルを量子化すると，メモリ使用量を削減できるものの，同時に特徴ベクトルの識別能力も低下するため，高い認識率を維持することが難しい．

上述の問題を解決する 1 つのアプローチは，ハッシュ表などのデータ構造を用い，特徴ベクトルの有無のみを記録することである [4]．このアプローチでは，特徴ベクトルを，距離のような量的な概念に基づく類似検索ではなく，ハッシュ値が同じかどうかという一致検索によって照合する．これにより，距離計算に必要な特徴ベクトルの情報を記録する必要がなくなり，大幅なメモリ削減が可能となる．しかし，この手法には，特徴ベクトルをハッシュ表にマッピングする際に偏りが生じるため，ハッシュ表の大半に何も記録されず，空間効率が悪いという問題点がある．

この問題を解決するために，我々は以前ハッシュ表の代わりに Bloomier filter [5] と呼ばれる空間効率の良い確率的データ構造を用いる手法を提案し，メモリ使用量の削減手法としての有効性を示している [6]．しかし，特定物体認識の従来法と比べ，どの程度有効性があるのかという疑問が残っており，検討が必要である．そこで本稿では，従来提案されている BoF(“bag of features”) モデルに基づく手法等と比較し，提案手法の有効性について検討する．また，提案手法で用いている認識の評価基準と物体認識率の関係を実験的に解析した結果について述べる．

2. 従来のメモリ削減手法

本節では，従来のメモリ削減手法として，BoF モデルに基づく手法 [3] とハッシュ表を用いた手法 [4] についてまとめておく．

2.1 BoF モデルに基づくメモリ削減法

本節では，BoF モデルに基づく特定物体認識の代表的な手法として VT(vocabulary tree) [3] を取り上げる．この手法は，階層的 k-means クラスタリングを行うことにより，VT と呼ばれる木を作成し，葉ノードを VW(visual word) とする手法である．具体的には，全ての物体から得られる特徴ベクトルに k-means クラスタリングを施して k 個のクラスタを作成し，各クラスタ重心を VT のノードとする．そして，クラスタ毎に対応付いている特徴ベクトルのみを用いて再び k-means クラスタリングを行い，クラスタを形成し，木を拡張していく．この処理を繰り返すことにより作成された VT の葉ノードを VW とする．この手法では，VT の枝の分岐数 x と，VT の深さ y を調節することにより，VW 数を変えることができる．例えば，$x = 10$，$y = 6$ のとき，VW の数は 10^6 となる．

認識処理では，木構造を根から葉に向かって調べることにより，検索質問ベクトルと最も近い VW を求める．この処理を繰り返し，tf-idf 重みを用いて VW の出現頻度の類似度を計算する．そして，最も類似度の高かった物体を認識結果とする．

この手法では，葉ノードの数を減らすと，メモリ使用量を削減できる．しかし，これは，多数の特徴ベクトルを一つのクラスタに対応付けることを意味するため，得られた VW は元の特徴ベクトルからかけ離れたものとなることがあり，識別性が低下するという問題が生じる．

2.2 ハッシュを用いたメモリ削減法

[4] の手法は，PCA-SIFT [1] で求めた 36 次元の特徴ベクトルを用いる特定物体認識手法であり，提案手法の基礎となる手法である．まず，特徴ベクトル $\boldsymbol{p}$ の第 1 次元から第 d 次元 ($d \leqq 36$) までをとり，

$$u_j = \begin{cases} 1 & \text{if } p_j \geqq 0 \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

によって 2 値化したビットベクトル $\boldsymbol{u} = (u_1, u_2, \ldots, u_d)$ を作成する．そして，

$$H_{\text{index}} = \left(\sum_{j=1}^{d} u_j 2^{(j-1)} \right) \mod H_{\text{size}} \tag{2}$$

というハッシュ関数より，ハッシュ表に登録するためのインデックスを求める．ここで，H_{size} は，ハッシュ表のサイズであり，現在 $H_{\text{size}} = 2^d$ としている．求まったインデックスに，特徴ベクトル $\boldsymbol{p}$ に対する物体 ID を登録する．登録時に衝突が生じた場合，リストとして登録していく．但し，計算コスト削減のために，リスト長が閾値 c より長くなる場合，リスト全体をハッシュ表から削除し，以後の登録を禁止する．

認識処理では，質問画像から PCA-SIFT で求めた特徴ベクトルを用い，データベース作成処理と同様に，ハッシュ表のインデックスを求める．そして，求めたインデックスに登録されている物体 ID の物体に対してスコアを増加させる．この処理を繰り返し，最終的に最もスコアの高い物体を認識結果とする．

以上の処理では，特徴ベクトルの変動によってインデックスが変化し，正しい物体に対応付かないという問題が生じる．この問題に対処するため，検索質問となる特徴ベクトル $\boldsymbol{q}$ において，各次元の値に許容変動幅 e を設け，$|q_j| \leqq e$ を満たす次元 j に対して，u_j だけではなく $u'_j = 1 - u_j$ も用いて特徴ベクトルを検索する．但し，処理時間を抑えるため，処理の対象とする次元数を閾値 b に限定する．$|q_j| \leqq e$ となる次元数が b 個より多くなる場合，高次元ものから b 個採用する．この手法では，多くの特徴ベクトルが得られる物体の方が，スコアを増加させる回数が多く高いスコアを得やすい．そこで，これを避けるため，次のようにスコアを増加させる．

$$s_i \leftarrow s_i + \frac{1}{\sqrt{r_i}} \tag{3}$$

ここで，s_i は物体 ID i の物体のスコア，r_i は物体 ID i の物体から得られる特徴ベクトル数である．

また，「認識に必要な特徴ベクトルの照合精度は画像によって異なる」という観点に基づき，処理を効率化するために，複数の識別器を直列に並べた構成とする．第 l 段では，$b = l - 1$ とした識別器を用いる．この識別器の多段階化による認識手法の特徴は，識別器の段が移ったとき，前の段で検索に用いられたビットベクトル以外のビットベクトルを用いて認識を行うとい

う，差分探索性があることである [4]．このため，最終段の識別器まで移ったとしても，最初から 2^b 個のビットベクトルを用いて認識するのとほぼ同じ処理時間で認識できる．

この手法では，d によってハッシュ表に必要なメモリ使用量が大幅に変化する．d を大きくすると，高い認識率を得ることができるが，ハッシュ表に必要なメモリ使用量が大きくなったり，大半が何も登録されず，空間効率が悪いという問題が生じる．一方，d を小さくすると，メモリ使用量を削減できるものの，異なる特徴ベクトルが同じビットベクトルに変換される確率が高くなるため，識別能力が失われるという問題が生じる．

以上の処理で識別能力が問題となる 1 つの原因は，ハッシュ表から検索された物体 ID 全てに対して式 (3) でスコアを増加させる点にある．正しい対応は 1 つであるため，良いものを選択する仕組みを導入すれば識別性が向上する. [4] では，そのような方式として，元の特徴ベクトルを保持しておき，ハッシュ表から検索された全ての特徴ベクトルのうち，検索質問の特徴ベクトルの最近傍となるものを選択してスコアを増加させる手法も提案している．この手法では特徴ベクトルを全て保持するためメモリへの負担は大きいが，精度が高まるという利点が得られる．本稿では，この手法も比較対象とする．

3. Bloom filter と Bloomier filter

本節では，提案手法に用いる Bloomier filter [5] と，その基礎となる Bloom filter [7] について述べる．

3.1 Bloom filter

Bloom filter は，ある要素とデータ集合が与えられたとき，この要素がデータ集合のメンバであるか否かを調べるために用いられる確率的データ構造である．Bloom filter は，集合のメンバではない要素が，集合のメンバであると判断される偽陽性 (false positive) が生じる可能性があるという問題点を持っている．一方，同じメモリ容量で保存できる要素数は，平衡二分木やハッシュ表と比べると圧倒的に多いという利点も持っている．本研究では，Bloom filter の空間効率の高さを利用し，特徴ベクトルの記録に用いる．本稿では，以後，データ集合の要素を特徴ベクトルとして話を進める．

図 1 に Bloom filter への特徴ベクトルの登録処理の流れを示す．まず，空の Bloom filter として，全ての bit を 0 に初期化してある m bit のビット列を用意する．ここで，m を Bloom filter の “TableSize”と呼ぶ．次に，特徴ベクトルをハッシュキーとし，k 個のハッシュ関数からそれぞれハッシュ値を得る．ここで得られるハッシュ値は 1〜m の整数値である．そして，得られた k 個のハッシュ値 $h_1, h_2, \ldots, h_k$ を基に，Bloom filter の $h_i (i = 1, 2, \ldots, k)$ 番目の bit を 1 にすることで特徴ベクトルの存在を登録する．特徴ベクトルの認識処理も登録処理と同様に，特徴ベクトルをキーとしてそれぞれのハッシュ関数からハッシュ値を得る．そして，得られたハッシュ値に対応する bit が全て 1 になっていれば，その特徴ベクトルはデータ集合に含まれていると判断する．

3.2 Bloomier filter

Bloomier filter は，複数の Bloom filter を組み合わせることで，登録した特徴ベクトルと関連する値を連想させることのできるデータ構造である．

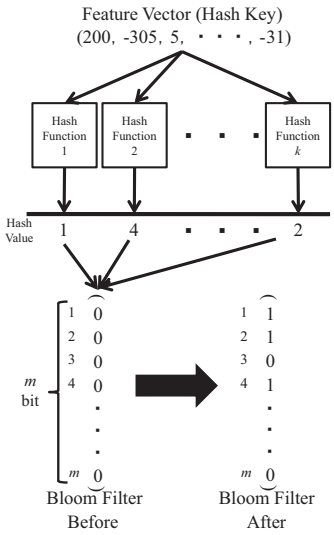


図 1　Bloom filter への特徴ベクトル登録処理の流れ

簡単のため，Bloomier filter で連想させる値が 0 と 1 の 2 種類のみの例について説明する．まず，2 つの Bloom filter $B^{(0)}$ と $B^{(1)}$ を用意する．次に，特徴ベクトルに連想させる値が 0(1) である場合，$B^{(0)}(B^{(1)})$ に特徴ベクトルを登録しておく．このとき，ある未知の特徴ベクトルが，$B^{(0)}(B^{(1)})$ に登録されているデータ集合のメンバならば，その特徴ベクトルから連想される値は 0(1) である可能性が高いと言える．本研究では，Bloomier filter を用いて，特徴ベクトルから物体 ID を連想させることを考える．Bloomier filter に任意の数字を連想させる場合，連想させる数字を二進数表現し，桁毎に 2 つの Bloom filter を用意すればよい．これにより，R 個の物体を識別するために必要な Bloom filter の数は，$2\log_2 R$ 個で済む．

4. 提 案 手 法

提案手法は，データベース作成に用いるデータ構造にハッシュ表ではなく Bloomier filter を用いる点が [4] と異なるが，他は同じである．以下では, [4] の手法との差分を中心に述べる．

4.1 データベース作成

提案手法では，物体 ID と特徴ベクトルに関連性を持たせるために，Bloomier filter を用いる．まず，物体 ID を $n$bit で表現するとし，$2n$ 個の Bloom filter を用意する．ここで，0 を連想させる Bloom filter を $B_1^{(0)}, B_2^{(0)}, \ldots, B_n^{(0)}$，1 を連想させる Bloom filter を $B_1^{(1)}, B_2^{(1)}, \ldots, B_n^{(1)}$ とする．それぞれの Bloom filter の TableSize $M_f^g (f \in \{1, 2, \ldots, n\}, g \in \{0, 1\})$ は，

$$M_f^g = a \times N_f^g \quad [\text{bit}] \tag{4}$$

とする．ここで，N_f^g は，物体 ID を bit 表現した際に f 番目の bit が g となる物体を考え，それらの物体から得られる特徴ベクトルの総数であり，a は TableSize の大きさを左右するパラメータである．予備実験より，a が小さいほどメモリ使用量を削減できるものの，認識率が低下することが分かっている．提案手法では，これを考慮し $a = 8$ とする．

次に，n 個の Bloom filter に特徴ベクトルを登録する．例えば，2bit で物体 ID を表現するとき，物体 ID の 3 を “10”と表現したとすると，1bit 目 (2bit 目) が “0”(“1”) であるため，$B_1^{(0)}(B_2^{(1)})$ にそれぞれ物体 ID 3 の物体から得られる特徴ベクトルを登録する．提案手法では，ハッシュ関数に MD5 を用い，得られる 128bit の bit 列を四等分 ($k = 4$) したものを Bloom filter への登録に用いるハッシュ値とする．

提案手法では，上記の処理で作成したデータベースに加え，誤りを検出するための Bloomier filter も作成する．方法は以下の通りである．まず，2 つの Bloom filter P_0, P_1 を用意する．そして，物体 ID を 2bit 表現したとき，1 の数が偶数個 (奇数個) の物体 ID から得られる特徴ベクトルを $P_0(P_1)$ に登録する．それぞれの Bloom filter の TableSize は，式 (4) と同様に得られる．

4.2 物体認識

提案手法の処理の流れを説明する．この手法では，質問画像から得られる特徴ベクトル $\boldsymbol{q}$ を用い，どの Bloom filter に $\boldsymbol{q}$ が含まれるかを調べ，物体 ID を求める．そして，求まった物体 ID の物体に対して式 (3) によりスコアを増加させる．但し，誤り検出 Bloomier filter で正しいと判断された物体 ID に対してのみ増加させる．この処理を質問画像から得られる特徴ベクトル全てに対して行い，最大スコアの物体を認識結果とする．以下に，具体的な処理について説明する．

まず，$\boldsymbol{q}$ を用い，物体 ID の $i(i = 1, 2, \ldots, n)$ 番目の bit が 0 か 1 かを求める．このとき，Bloom filter $B_i^{(0)}(B_i^{(1)})$ に登録されていれば，物体 ID の i 番目の bit を 0(1) とする．$B_i^{(0)}$ と $B_i^{(1)}$ のいずれにも登録されていない場合，この特徴ベクトルはデータベースに登録されていないとし，i 番目以降の処理を打ち切る．この処理では，$B_i^{(0)}$ と $B_i^{(1)}$ の両方に登録されていた場合に問題が起こる．これはどちらかの Bloom filter が偽陽性を引き起こし，実際には登録されていないにもかかわらず，登録されていると判断されるためである．提案手法では，このような場合両方の可能性を試すことにより対処する．具体的には，i 番目の bit が 0 となる物体 ID と 1 となる物体 ID の両方についてスコアを増加させる．但し，全ての bit に対して上記の処理を行うと，誤ってスコアを増加させる回数が増えるため，提案手法では，偽陽性に対処する物体 ID の桁数を閾値 t 個に限定する．偽陽性に対処する桁数が t 個以上の場合，その特徴ベクトルは認識には寄与しないと考え，スコアを増加させない．

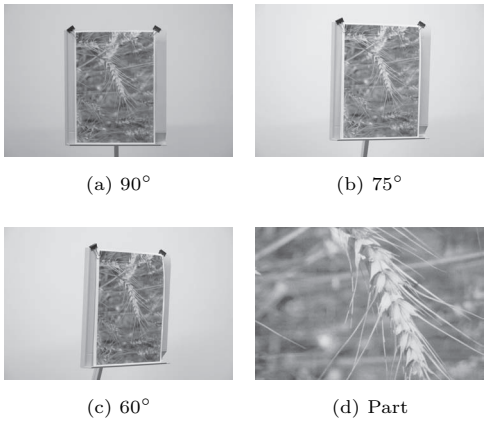
上記の処理により，求まった物体 ID の物体に対して，誤り検出 Bloomier filter を用い，実際にスコアを増加させるかどうかを決める．このとき，誤り検出 Bloomier filter に登録されていなかった場合や，P_0, P_1 の両方に登録されていた場合，求まった物体 ID は誤りと判断し，破棄する．この処理を質問画像から得られる特徴ベクトル全てに対して行い，最終的に最もスコアの高い物体を認識結果とする．

5. 実　　験

5.1 実験条件

本研究では，提案手法の有効性を確かめるために，1 万個の 2 次元物体に対して実験を行った．まずデータセットについて説明する．図 2 に物体の例を示す．データベース作成用画像から得られる特徴ベクトルの総数は，約 2 千万個である．質問画像としては，データベースに含まれる画像から合計 500 枚を無作為に選択し，図 3 に示すように 4 通りに撮影した画像を用いた．質問画像の数は，$500 \times 4 = 2{,}000$ 枚で，1 枚の質問画像から得られる平均特徴ベクトル数は約 630 個である．

図 2　2 次元物体の例



(a) 90°　(b) 75°　(c) 60°　(d) Part

図 3　検索質問の例

本実験では，以下の 3 手法を比較に用いる．すなわち, [3] で提案されている手法 VT, [4] で提案されている特徴ベクトルの距離計算をしない手法 (ハッシュ(距離計算なし))，ならびに距離計算を行う手法 (ハッシュ(距離計算あり)) である．本実験では，VT で用いる正規化には，L_2-ノルムを用い，ハッシュ(距離計算あり・なし) で用いるハッシュ表のサイズを $H_{\text{size}} = 2^d$ とした．実験に用いた計算機は，CPU が AMD Opteron8378 2.4GHz，メモリが 128GB のものである．ここで，以下の実験結果で示す処理時間は，1 枚の画像の認識にかかった平均の処理時間であり，特徴ベクトルの抽出時間等は含まない．ハッシュに基づく比較手法および提案手法で用いるパラメータ d, b，e，c を確認しておく．ビットベクトル変換に用いる次元数 d，ビットベクトルで反転の操作を行う次元数 b，値の変動幅 e，重複するビットベクトルの上限数 c の 4 つである．本実験では，$d = 22$，24，26，28，30，$b = 0$，1, . . . ，10，$e = 200$，$c = 1$，2，. . . , 10 とした．また，提案手法ではこれらに加えて偽陽性に対処する bit 数 t を用いており，$t = 0$，1，2 とした．VT で用いたパラメータは表 1 に示すものである．

5.2 実験結果

図 4 と図 5 に，認識率毎に最もメモリ使用量の少なかったときの処理時間と，そのときのメモリ使用量をそれぞれ示す．

表 1　実験に用いた VT(Vocabulary Tree) のパラメータ

分岐数 x	深さ y	分岐数 x	深さ y
10	4	11	6
	5	12	
	6	13	
	7	14	

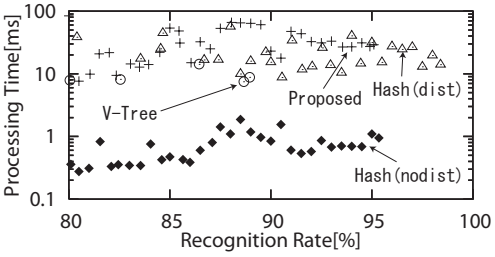


図 4　認識率と処理時間の関係と従来手法との比較

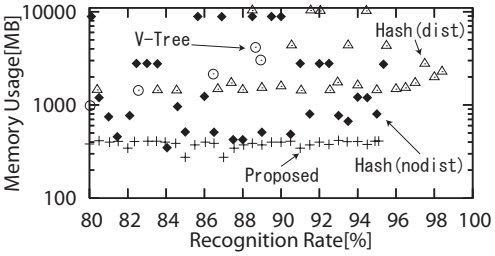


図 5　認識率とメモリ使用量の関係と従来手法との比較

図 4 より，認識率が同程度のとき，提案手法は他の手法と比べて最も処理時間がかかることが分かった．これは，物体 ID を求めるために，提案手法では $2n$ 個の Bloom filter を調べなければならないためである．次に，図 5 に着目すると，提案手法は，ハッシュ(距離計算あり) と比べ，認識率では劣るものの，メモリ使用量を削減できることが分かった．

提案手法で最も認識率の高かった結果と，3 手法の結果のうち，それぞれ提案手法と同程度の認識率になるものと，最も認識率の高かったものを表 2 にまとめる．ここで，提案手法と同程度の認識率となる結果を “同認識率”，最も認識率の高かった結果を “最大認識率”と表記する．結果より，認識率が同程度のとき，提案手法が最も少ないメモリ使用量で物体を認識できることが分かった．特に，VT と比較すると，提案手法の方が処理時間がかかるものの，認識率，メモリ使用量の点で大きく有効性があると言える．VT で誤認識した物体の特徴として，得られる特徴ベクトルの数が少ないことが挙げられる．この特徴より，得られる特徴ベクトルの数が少ないにもかかわらず，複数の特徴ベクトルをクラスタリングなどによって 1 つの VW にまとめてしまうことが，認識率低下の原因と考えられる．このことから，特定物体の認識には，ある程度元の特徴ベクトルを保存しておく必要があると言える．また，ハッシュ(距離計算あり) と比較すると，認識率，処理時間が同程度でメモリ使用量を 1/3 以下にすることができた．

以上の結果をまとめると，若干の認識率の低下が許容できるのであれば，提案手法を用いるとメモリ使用量を抑えることができると言える．

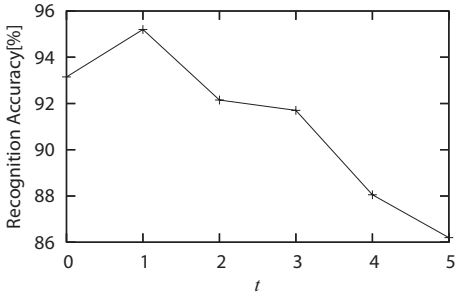


図 6　偽陽性に対処する桁数 t と認識率の関係

5.3　提案手法に用いるスコアと認識率の関係

提案手法では，特徴ベクトルの照合に距離といった量的な概念に基づく類似検索ではなく，登録されている Bloom filter が同じかどうかという一致検索を用いている．このような単純な照合にもかかわらず，95%以上の認識率を得ることができる．提案手法では，特徴ベクトルの照合結果より求まった物体に対して式 (3) のようにスコアを増加させ，最終的に最大スコアを持つ物体を認識結果としている．そこで本研究では，提案手法で用いるスコアに着目し，提案手法が高い物体認識率を得る理由について調べる．提案手法では，偽陽性に対処する桁数 t や，ビットベクトルの反転操作を行う次元数 b によって，検索質問ベクトル $\boldsymbol{q}$ 毎に複数の物体 ID を求め，対応する物体のスコアを増加させる．しかし，ある $\boldsymbol{q}$ に対する正解物体は本来 1 つである．そこで本節では，複数の物体に対してスコアを増加させることが物体の認識率に与える影響について述べる．実験では，簡単のため「正しい物体に対してスコアを増加させる可能性は画像によって異ならない」という仮定の下，認識器の多段階化を用いない．

5.3.1　偽陽性に対処する桁数 t と認識率の関係

まず偽陽性に対処する桁数 t が認識率に与える影響から見ていく．図 6 に最も認識率の高かったパラメータ $b = 7, c = 4, d = 30$ を用い，t を $t = 0, 1, \ldots, 5$ と変化させた場合の認識率を示す．図 6 より，t を大きくすれば，認識率が下がるということが分かる．これは，正解物体のスコアを増加させる可能性は高くなるものの，それ以上に正解物体以外のスコアを増加させる可能性も高くなるためである．また図 6 より，$t = 1$ の時，最も認識率が高いことから，提案手法ではほとんど偽陽性に対処する必要がないと言える．

5.3.2　ビット反転の操作を行う次元数 b と認識率の関係

次に，ビットベクトルの反転操作を行う次元数 b が認識率に与える影響について見ていく．本研究では，提案手法を物体 ID 検索器に見立て，物体 ID 検索の再現率 (Recall) と適合率 (Precision) の 2 つの指標を用いて b の影響を解析する．ここで再現率は，全検索質問ベクトル $\{\boldsymbol{q}\}$ のうち，提案手法によって検索された物体 ID に，正しい物体 ID が含まれている $\boldsymbol{q}$ の割合を意味する．また適合率は，$\boldsymbol{q}$ に対して検索された物体 ID に含まれている正しい物体 ID の割合を意味する． 図 7，8 に最も認識率高かったパラメータ $c = 4, d = 30, t = 1$ を用い，b を変

表 2　各手法の性能評価

手法	パラメータ	認識率 [%]	処理時間 [ms]	メモリ使用量 [MB]
提案手法	$b=7,\ c=4,\ d=30,\ e=200,\ t=1$	95.10	29.01	411
VT(同認識率)	$x=14,\ y=6$	88.95	8.75	3017
VT(最大認識率)	同上	同上	同上	同上
ハッシュ(距離計算なし)(同認識率)	$b=5,\ c=9,\ d=24,\ e=200$	95.15	1.00	799
ハッシュ(距離計算なし)(最大認識率)	$b=6,\ c=2,\ d=28,\ e=200$	95.35	0.95	2736
ハッシュ(距離計算あり)(同認識率)	$b=5,\ c=4,\ d=22,\ e=200$	95.10	30.76	1439
ハッシュ(距離計算あり)(最大認識率)	$b=7,\ c=9,\ d=24,\ e=200$	98.4	13.99	2275

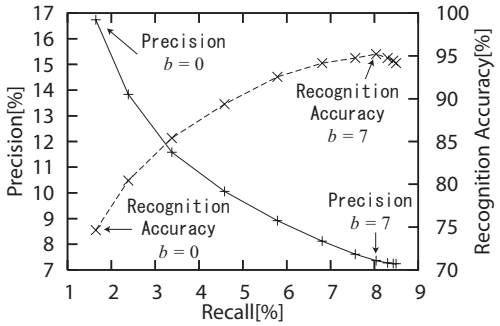


図 7　再現率と適合率および認識率の関係

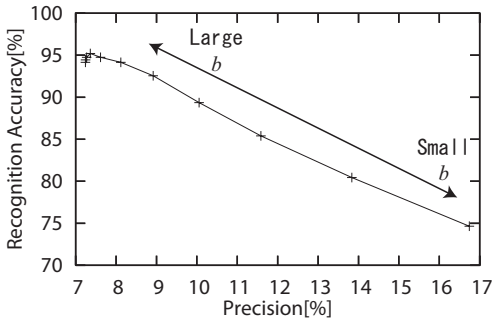


図 8　適合率と認識率の関係

化させた場合の再現率，適合率，認識率の関係を示す．図 7 より，b を大きくすると再現率が高くなる，つまり正解物体の物体 ID が検索される可能性が高くなることが分かる．また，再現率が高くなると正解物体のスコアを増加させる回数が増えるため，正解物体のスコアが最大スコアとなる可能性も高くなる．このため，b を大きくすると認識率が高くなると言える．しかし，b を大きくし過ぎると，正解物体以外のスコアの方が高くなり，若干認識率が下がる．これより，b を大きくするには限界があると言える．また図 7 より，b を大きくすると再現率は高くなるものの，正解物体以外の物体 ID が検索される可能性も高くなるため，適合率は低下している．

次に図 8 に着目すると，b を大きくすると適合率は低下するが，認識率は高くなることがわかる．これは，正解物体以外の物体 ID が多く検索されても，物体の認識率にあまり影響を与えないことを意味する．具体的に見ていこう．検索される正解物体以外の物体 ID の個数は，検索質問画像当たり平均で $b=0$ の時 44 個，$b=7$ の時 1411 個である．このとき，検索された物体 ID 毎に，検索質問画像当たり平均で何個検索されているか調べたところ，表 3 のような結果を得た．これより，正解物体以外の物体 ID が多数検索されても，ある 1 つの物体 ID が

表 3　物体 ID の平均検索個数

	$b=0$	$b=7$
正解物体	11	55
正解物体以外	1	1

集中して検索されないため，提案手法のようなスコアを用いると高い認識率を得ることができる．

以上の結果より，提案手法を物体 ID 検索器と見立てた場合，適合率よりも再現率を重視した方が，高い物体の認識率を得ることができると言える．

6. おわりに

本稿では，Bloomier filter を用い，特定物体認識に必要なメモリ使用量を削減する手法を提案した．1 万個の平面物体を用いた実験の結果，従来のメモリ削減手法，および特徴ベクトルの距離計算に基づいて照合を行う手法と比較し，提案手法は同程度の認識率を得るのに必要なメモリ使用量を削減できることが分かった．また，提案手法のような一致検索に基づく物体認識手法では，特徴ベクトルの変動に対処し，検索のための特徴ベクトルの数を増やせば，認識率が向上することを示した．

今後の課題として，認識率の向上や更なる大規模な物体データベースを用い，提案手法のスケーラビリティを調べることなどがある．

謝辞　本研究の一部は科学研究費補助金基盤研究 (B)(19300062) の補助による．

文　献

[1] Y. Ke and R. Sukthankar: “PCA-SIFT:A more distinctive representation for local image descriptors”, Proc. of CVPR2004, Vol. 2, pp. 506–513 (2004).
[2] 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, pp. 1135–1143 (2009).
[3] D. Nistér and H. Stewénius: “Scalable Recognition with a Vocabulary Tree”, Proc. CVPR2006, pp. 775–781 (2006).
[4] 野口和人, 黄瀬浩一, 岩村雅一：“近似最近傍探索の多段階化による物体の高速認識”, 画像の認識・理解シンポジウム (MIRU2007) 論文集，OS-B2-02, pp. 111–118 (2007).
[5] B. Chazelle, J. Kilian, R. Rubinfeld and A. Tal: “The Bloomier Filter: An Efficient Data Structure for Static Support Lookup Table”, Proc. 15th Annual ACM-SIAM SODA, pp. 30–39 (2004).
[6] 井上勝文, 黄瀬浩一：“Bloomier filter を用いた特徴ベクトルの圧縮表現とその特定物体認識への応用”, 画像の認識・理解シンポジウム (MIRU2009) 論文集，OS12-2, pp. 343–350 (2009).
[7] B. H. Bloom: “Space/Time Trade-offs in Hash Coding with Allowable Errors”, Commun. ACM, Vol. 13, No. 7, pp. 422–426 (1970).