

局所記述子に基づく物体認識のためのメモリ削減の実験的検討

局所記述子のビット削減によるアプローチ

野口 和人[†] 黄瀬 浩一[†] 岩村 雅一[†]

[†] 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: noguchi@m.cs.osakafu-u.ac.jp, {kise,masa}@cs.osakafu-u.ac.jp

あらまし 局所記述子を用いた物体認識において、認識率、処理時間、メモリ量について良いバランスを実現することは重要な問題である。従来、局所記述子をベクトル量子化して visual word を得、それを認識に用いる手法が多数提案されている。インスタンス (特定物体) を対象とした認識では、数多くの visual word を用いる手法は高い認識率が得られるものの、処理時間やメモリ量に問題が生じる。一方、局所記述子を量子化せずそのまま用いる手法であっても、近似最近傍探索によって処理が大幅に短縮できることが知られている。本稿では、ベクトル量子化を行わない場合でも、メモリ量の問題を解決する手法として、特徴ベクトルの各次元を記録するビット数を削減する手法を提案する。10 万画像のモデルを用いて認識実験を行った結果、各次元を 2 ビットで表現したとき、ビット数削減前 (16 ビット) と比べて、認識率は少し下がるものの、処理時間は殆ど変化しないことが分かった。

キーワード 物体認識, 局所記述子, 近似最近傍探索, メモリ容量, スカラー量子化, ビット削減

Experimental Evaluations of a Memory Reduction Method for Object Recognition Based on Local Descriptors

An Approach by Bit Reduction of Local Descriptors

Kazuto NOGUCHI[†], Koichi KISE[†], and Masakazu IWAMURA[†]

[†] Graduate School of Engineering, Osaka Prefecture University

1-1 Gakuencho, Naka, Sakai, Osaka, 599-8531 Japan

E-mail: noguchi@m.cs.osakafu-u.ac.jp, {kise,masa}@cs.osakafu-u.ac.jp

Abstract For object recognition based on local descriptors, balancing the recognition rate, processing time and memory requirement is an important issue. In conventional methods, local descriptors are vector-quantized to “visual words” for describing images. Although many visual words allow us to improve the recognition rate for recognizing instances (specific objects), they also pose a problem of processing time and memory requirement. On the other hand, even if a recognition method employs local descriptors without vector quantization, approximate nearest neighbor search enables us to cut the processing time drastically. In this report, we propose a method of memory reduction applicable even to methods without vector quantization. The proposed method reduces the memory requirement by limiting the number of bits for the record of each dimension of feature vectors. From experimental results on 100,000 images, it has been confirmed that a representation with 2 bit / dimension subtly decreases the recognition rate, but hardly changes the processing time.

Key words Object recognition, Local descriptors, Approximate nearest neighbor search, Required memory, Scalar quantization, Bit reduction

1. ま え が き

SIFT(Scale-Invariant Feature Transform) [1] などの局所記

述子を用いると、隠れや照明条件の変動に比較的頑健な物体認識が実現可能であるため、現在、注目を集めている [2]. 認識の基本は、“Bag of Words” あるいは “Bag of Features” と呼ば

れるモデルであり、局所記述子の配置や共起を考慮せず、その頻度のみを手がかりとして物体を認識するものである。本稿では物体認識の中でも特にインスタンス(特定物体)の認識に焦点をあてて議論していく。

一般に、画像から抽出される局所記述子の数は、数百から数千、場合によっては数万となるため、局所記述子の照合に必要な処理時間や、記憶に必要なメモリ容量は膨大となる。したがって、認識精度を一定レベルに保ったまま、いかに処理時間やメモリ容量を削減するかが重要な研究課題となっている。

この問題に対して、従来、多くの手法では、モデル作成用の画像から得た局所記述子をベクトル量子化して数千から数十万程度の visual word を定め、それを用いて画像を記述するというアプローチを採用している[3]。未知の画像を認識する際には、その画像から得た局所記述子を visual word に変換し、頻度などを計測することになる。このようなアプローチでは、visual word の数が十分少なければ高速な処理が期待できる。しかし一方で、特定物体の認識においては、visual word の数が多くなれば、十分な認識率を達成できないことが指摘されている[4]。visual word の数が増えればそれだけ、ベクトル量子化に必要な計算時間を無視することができず、また、visual word 自体の記録にもメモリ容量の面で問題が生じる。

以上の利点・問題点は、極端な場合、すなわち、モデル作成用の画像から得た個々の局所記述子をそのまま visual word とする場合に、最も顕著になる。例えば、VGA サイズの一般的な画像からは、2千程度の局所記述子が抽出される。従って、VGA サイズの10万画像をモデル作成に用いる場合、visual word の数は2億となり、照合ならびに記憶に膨大な計算資源が必要となる。一方で、大量の局所記述子をモデルに用いることにより、高精度な認識を実現することが可能となる。

処理時間の問題に対する一つの解決策は、局所記述子の照合に「近似最近傍探索」を導入することである[5]。これによって、例えば上記の規模の認識タスクの場合、単純な全数照合の場合と比べて、認識率を殆ど低下させることなく、処理速度を 10^{-6} 未満にできることが知られている。一方、メモリ容量の問題については、ベクトル量子化を粗くすることが解決策の一つであるが、これは認識率の低下を意味するため、必ずしも得策ではない。

そこで本稿では、この問題に対する新しい解決策と、その実験的検討の結果を報告する。解決策としては、ベクトル量子化により得られる visual word を絞り込むというアプローチではなく、個々の局所記述子の記録に必要なメモリ量を削減するというアプローチをとる。具体的には、局所記述子の特徴ベクトルについて、記録に必要な各次元のビット数を、実数や整数に必要な容量(16~32ビット)から削減することを試みる。これは、局所記述子にスカラー量子化を適用することと見なすことができる。個々の特徴ベクトルに必要なメモリ容量を削減することによって、多数の局所記述子(あるいは visual word)を記録する場合でも、十分な削減効果を見込むことができる。特徴ベクトルを少ないビット数で表現する場合、認識率の低下など悪影響が出ることが予測される。この点について、10万画像を

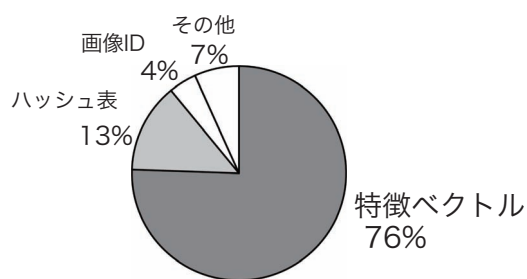


図1 メモリ使用量の内訳(10万画像のデータベース)

用いて実験的に検討した結果、特徴ベクトルの各次元を記録するビット数を2ビットとしても、認識率は殆ど変化しないことが分かった。

2. 近似最近傍探索の従来法

提案手法の詳細について述べる前に、近似最近傍探索の従来法についてまとめておく。ここで取り上げる従来法は、比較実験に用いるものである。

最近傍探索で最も時間がかかるのは距離計算である。最近傍探索の高速化には、個々の距離計算自体を高速に行う方法と、距離計算の対象を効率よく絞り込む方法の2通りがある。近似最近傍探索では、主に後者の絞り込みを大胆に行うことで、処理時間を削減する。ところがその代償として、最近傍が距離計算の対象から外れてしまい、求まらない可能性がある。どの程度の近似が適切であるのかは、対象とするタスクに依存するため、近似最近傍探索の手法では、近似の程度を調節するためのパラメータを設けている。以下では、近似最近傍探索の代表的な手法として ANN について述べる。

ANN(Approximate Nearest Neighbor)[6]は、2分木を用いて近似最近傍探索を高速に行う手法である。木のノードは、特徴空間を分割した hyperrectangle に対応しており、葉ノードには単一の特徴ベクトルが対応付けられている。ANNでは、木構造の探索によって距離計算の対象となる特徴ベクトルを収集し、その中で距離が最短のものを近似最近傍探索の結果として出力する。ANNには近似の程度を表すパラメータとして許容誤差 ϵ がある。 ϵ が大きければ、より大幅な近似を行って対象となる特徴ベクトルを絞り込むため、処理時間が短縮できる。

3. 提案手法

3.1 考え方

局所記述子を用いて認識を行う従来手法の問題点は、特徴ベクトルの数が多いため、大規模な認識を行う場合にメモリ使用量と処理時間が膨大になってしまうというものである。

処理時間の問題に対して、野口らは近似最近傍探索による識別器を多段階に接続することで処理時間を削減する手法[5]を提案している。しかし、この手法ではメモリ使用量の問題に触れていない。この手法のメモリ使用量の内訳を図1に示す。図より約8割が特徴ベクトルを保持するために使われていることがわかる。そこで、提案手法では野口らの手法を基本とし、特徴ベクトルを量子化して少ないビット数で表現することで、メ

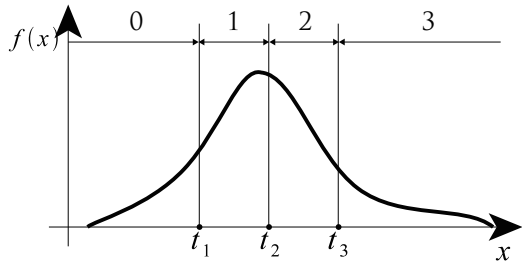


図 2 スカラー量子化

メモリ使用量を削減することを試みる。

ベクトルを量子化する方法には、代表的なものとしてベクトル量子化とスカラー量子化がある。ベクトル量子化は、あらかじめ、ある個数の代表ベクトルを求めておき、入力ベクトルを最近傍の代表ベクトルの符号に置き換えるものである。ベクトル量子化は、局所記述子を用いた認識手法で多く使われている[3]。しかし、大規模なデータに対して量子化誤差の少ない代表ベクトル(前述の visual word)を効率よく求めることは難しい。また、代表ベクトルの数が増えると、入力ベクトルの最近傍の代表ベクトルを探索する処理に時間がかかるという問題点がある。一方、スカラー量子化は、入力ベクトルの次元ごとに量子化を行うものである。同じ符号数で比較した場合はベクトル量子化と比べて量子化誤差が大きくなる。しかし、量子化の際に最近傍探索が必要ないため処理時間は少なく済むと考えられる。そこで、提案手法では処理時間を重視してスカラー量子化を用いることとする。

量子化には、メモリ使用量を削減できるという利点ばかりではなく、認識率が低下する可能性があるという問題点もある。これは、もとは別々の特徴ベクトルであったものが、量子化により同じ特徴ベクトルになり識別性が低下するからである。特徴ベクトルの識別性の低下が認識率に及ぼす影響は、提案手法の認識結果が投票によって決定されるため、簡単にはわからない。これは、正解の得票数が逆転しなければ、誤った票が他の画像に入っても誤認識は生じないためである。そこで、量子化のビット数と認識率の関係については実験で検証する。

3.2 スカラー量子化

まず、提案手法で用いるスカラー量子化について説明する。スカラー量子化では、次元ごとに量子化を行う。各次元 2bit で量子化する場合を図 2 に示す。 $f(x)$ は特徴ベクトルのある次元の分布である。 t_i は量子化の閾値であり、例えば $-\infty$ から t_1 の範囲のものは 0 に符号化する。 t_i の値については、次式を満たすように次元ごとに定める。

$$\int_{-\infty}^{t_1} f(x)dx = \int_{t_1}^{t_2} f(x)dx = \int_{t_2}^{t_3} f(x)dx = \dots \quad (1)$$

3.3 ハッシュ関数

野口らの手法と同様、提案手法でも、ハッシュ関数を用いてデータをハッシュ表に登録し、検索に用いる。ハッシュ表にデータを登録、検索する場合には、ハッシュ関数により処理をおこなうインデックスを知る必要がある。ここでは提案手法で用いるハッシュ関数について説明する。

PCA-SIFT によって得られる 36 次元の実数値ベクトル x は、主成分分析の結果であるので、前方の次元の方が固有値が大きいという性質がある。そこで、 x の第 1 次元から第 d 次元までを取り、 $\hat{x} = (x_1, x_2, \dots, x_d)$ とする。次に、

$$u_j = \begin{cases} 1 & \text{if } x_j - \mu_j \geq 0, \\ 0 & \text{otherwise,} \end{cases}$$

を用いて次元ごとに二値化を行いビットベクトル $u = (u_1, \dots, u_d)$ を作成する。ここで μ_j は、画像データベース中すべての x_j の平均値である。そして、

$$H_{\text{index}} = \left(\sum_{i=0}^d u_i 2^i \right) \bmod H_{\text{size}} \quad (2)$$

によってハッシュ値を求める。ここで H_{size} は、ハッシュ表のサイズである。

3.4 データ登録

特徴ベクトルをハッシュ表に登録するためには、前述のハッシュ関数によりハッシュ値を求める必要がある。ハッシュ表には、画像 ID とともに量子化によってデータ量を削減した特徴ベクトルを登録する。登録時に衝突が生じた場合は、複数の特徴ベクトルをリストとして登録する。このとき、リストが長くなりすぎると、検索の際に距離計算のコストがかかりすぎるといった問題が生じる。そこで本手法では、リスト長 n に対する閾値 c を設け、 $n > c$ を満たすとリスト全体をハッシュ表から削除する。同じハッシュ値を持つ特徴ベクトルが多いということは、その特徴ベクトルが画像の識別にあまり寄与しないことを意味する。従って、削除しても影響は比較的少ないと考えられる。

以上の処理を、データベースに登録する全ての特徴ベクトルに対して施すことにより、データの登録は完了する。

3.5 検索

次に検索について述べる。本手法では、検索質問 q から得た各特徴ベクトル q に対して、ハッシュ関数を適用しハッシュ表から特徴ベクトルを検索する。ここで得られた特徴ベクトルの集合を X とする。次に、 q を量子化したベクトルと、 X に含まれるベクトルとのユークリッド距離を計算し、最近傍となる特徴ベクトル x_* を求める^(注1)。

そして、 x_* に対応する画像 ID に投票する。もし、最近傍となる特徴ベクトルが複数ある場合には、それらすべてに対して投票処理を施す。検索質問のすべての特徴ベクトルに対してこの処理を行い、最終的に最も得票数の多いものを回答とする。

この処理において、最も重要なステップは、いかに q に対する特徴ベクトルを検索するかにある。最も単純な手法は、登録時と同様に q に対してもビットベクトルを求め、ハッシュ関数によって同じハッシュ値を持つ特徴ベクトルを求めることで

(注1): 検索時に q に対して量子化を行わない方法も考えられる。この場合、データベースから得た特徴ベクトル X を代表値に変換し q との距離を計算する。ただし、この場合は距離計算のたびに代表値への変換が必要であるため、処理時間が 1.5 倍ほど必要であった。

ある．ところが，このような処理では，距離の計算回数は十分削減できるものの，次の理由によって十分な認識率を得ることができない．それは，特徴ベクトルの各次元の値が撮影条件によって変動するためである．もし，閾値を超えるような変動があると，ビットベクトルが異なるものとなり，もはや対応する特徴ベクトルを得ることができなくなる．本手法では，値の変動幅 e をパラメータとして，変動への対処を施す．具体的には， $q = (q_1, \dots, q_d)$ とするとき， $|q_j - \mu_j| \leq e$ を満たす次元 j に対しては， u_j だけではなく $u'_j = (u_j + 1) \bmod 2$ (0 ならば 1, 1 ならば 0) も用いて，特徴ベクトルを検索する．ただし，このような「両方試す」という処理を制限なく導入すると，膨大な計算時間が必要となってしまう．この処理では，処理の対象となる次元数を b とすると， 2^b 通りのビットベクトルを用いてハッシュ表にアクセスすることになる．そこで本手法では， b をあまり大きくない値に留めることとする． $|q_j - \mu_j| \leq e$ を満たす次元の数が b を上回るときには，次元のインデックスが大きいものから b 個を採用する．

4. 実験

4.1 実験条件

実験では，近似最近傍探索の従来法である ANN^(注2)を用いた認識手法と提案手法を比較した．

実験には以下に述べる画像データベース，検索質問画像を用いた．局所記述子としては，PCA-SIFT^(注3)を用いた．PCA-SIFT では 36 次元の特徴ベクトルが得られる．ハッシュ表のサイズは $H_{\text{size}} = 2^d$ とした．以下に示す処理時間は，検索質問の画像 1 枚あたりの認識に要した時間を表す．ただし，特徴ベクトルの抽出に必要な時間は含まない．使用計算機は，CPU が AMD Opteron 2.8GHz，メモリ 32GB のものである．

4.1.1 画像データベース

実験に用いた画像について説明する．まず，収集方法の異なる A, B, C の 3 種類のデータセットを準備した．A は，Google のイメージ検索を用いて収集した 3,100 枚の画像である．検索キーワードとしては，“ポスター”，“雑誌”，“表紙”などを用いた．図 3(a) に例を示す．B は PCA-SIFT のサイトで公開されている画像であり，画像数は 18,500 枚である．このデータは主に自然写真や人物の写真などで構成されている．図 3(b) に例を示す．C は，写真共有サイトの flickr において “animal”，“birthday”，“food”，“japan” などのタグにより収集した 78,400 枚の画像からなる．主に図 3(c) に示すような物体や自然の写真，人物の写真などを含む．なお，収集の際には，600×600 pixel 以下のサイズの画像は除外し，画像の長辺が 640 pixel 以下になるように縮小した．また，特徴ベクトルが 100 個以下の画像も除外した．画像の一辺の長さの平均は A, B, C それぞれ 498, 612, 554 pixel であった．

次に，A, B, C の画像を用いて，表 1 に示した画像数からな

表 1 データベースに含まれる画像数

登録画像数	1,000	5,000	10,000	50,000	100,000	
内訳	A	334	1,667	3,100	3,100	3,100
	B	333	1,667	3,450	18,500	18,500
	C	333	1,666	3,450	28,400	78,400

る 5 種類のデータベースを作成し実験に用いた．ここで，大きいデータベースは，小さいデータベースをその一部として含む．1 万枚の画像データベースの特徴ベクトルの各次元の分布 $f(x)$ を図 4 に示す．1 次元目は双峰性の分布であり，2 次元目以降は単峰性の分布を示す．また，次元が大きくなるにつれて分散が小さくなる．平均値はいずれも 0 の付近である．また，特徴ベクトルの個数は画像 1 枚あたり平均 2000 個であった．実験では，どの枚数のデータベースを用いる場合でも，1 万枚の画像データベースから得られた $f(x)$ を量子化に用いた．

4.1.2 検索質問画像

検索質問としては，データセット A, B, C のそれぞれから 100, 200, 200 枚の合計 500 枚を無作為に選択した．次に，これらを A4 の用紙に印刷し，カメラを用いて撮影した．得られた画像の例を図 5 に示す．図に示すとおり，紙面全体が写る配置で，紙面に対するカメラの光軸の角度 θ を $90^\circ, 75^\circ, 60^\circ$ に変化させた．また，角度を 90° として紙面の一部を撮影した．その結果，1 枚の紙面に対して，合計 4 通りの画像を得た．さらに，撮影した画像を 512×341 pixel に縮小し，PCA-SIFT により特徴ベクトルを求めた．その結果，画像 1 枚あたり平均 612 個の特徴ベクトルが得られた．

4.2 特徴ベクトルのメモリ容量と認識率

まず，量子化によりデータ量を削減すると，どの程度認識率に影響がでるか調べた．結果を図 6 に示す．横軸は特徴ベクトルの各次元の量子化ビット数を表している．たとえば，2bit の場合には，PCA-SIFT の特徴ベクトルが 36 次元なので，一ベクトルあたり $2\text{bit} \times 36 = 72\text{bit}$ 必要となる．ここで，16bit と 0bit の場合は少し特殊であることを述べておく．16bit の場合には量子化を行わず元のデータをそのまま用いた（野口らの手法）．0bit の場合には，距離計算を行うための特徴ベクトルが存在しないため，ハッシュ表から得られた集合 X のすべてに対して投票を行った．登録画像数は 1000, 10000, 100000 枚，パラメータは 16bit の場合により結果の得られた $b = 10, c = 10, d = 28, e = 400$ について調べた．また，メモリ使用量は 100000 枚のデータベースについてのみ調べた．

グラフから，2bit まで削減しても，認識率に大きな影響は見られないことがわかる．1bit の場合には登録画像数が多いほど認識率の低下が著しいことがわかる．これは，登録されている特徴ベクトルの数が増えるほど，識別性の不足が問題となるためと思われる．0bit の場合には認識率が大きく低下することがわかる．以上より，各次元 2bit，つまり特徴ベクトル 1 つあたり 9 バイトで表現できることがわかる．各次元を 16bit で表現した場合と比べると，特徴ベクトル単体で $1/8$ の容量であり，実際のメモリ使用量も約 $1/3$ と大幅な圧縮が可能である．以降の実験では各次元 2bit で量子化を行った場合の詳しい性質につ

(注2): ANN としては <http://www.cs.umd.edu/~mount/ANN/> で提供されているソースコードを用いた．

(注3): <http://www.cs.cmu.edu/~yke/pcasift/> で提供されている．



(a)



(b)



(c)

図 3 登録画像の例

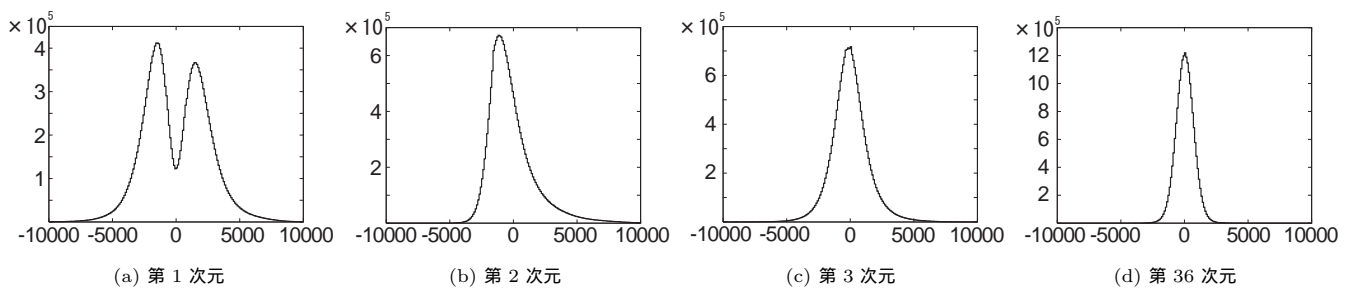


図 4 特徴ベクトルの値分布．横軸は各次元の値，縦軸は頻度．

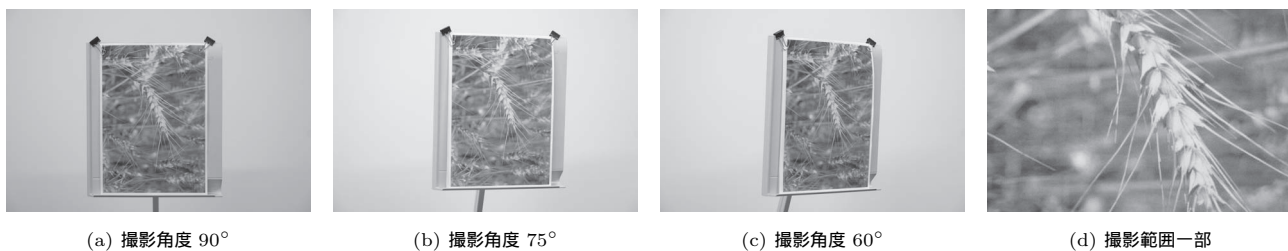


図 5 検索質問の例

いてみていく．

4.3 2bit と 16bit で量子化する場合の比較

次に，提案手法の 2bit と 16bit の場合について登録画像数と認識率と処理時間の関係を比較した．

登録画像数を 5000 枚から 10 万枚まで変化させ，実験を行った．パラメータは先ほどと同様に $b = 10, c = 10, d = 28, e = 400$

を用いた．登録画像数を变化させた結果を図 7 に示す．

グラフより，認識率は 0.3% ほど低下することがわかる．認識率の低下は，距離計算の結果が量子化誤差によって変化したためと思われる．それは，ハッシュ関数は特徴ベクトルの量子化と無関係なため，16bit の場合でも 2bit の場合でも，距離計算の対象となるベクトルは変化しないからである．

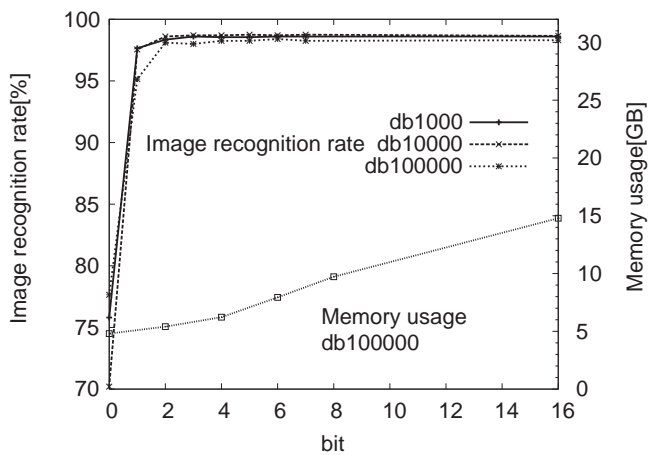


図 6 特徴ベクトルの容量と認識率の関係

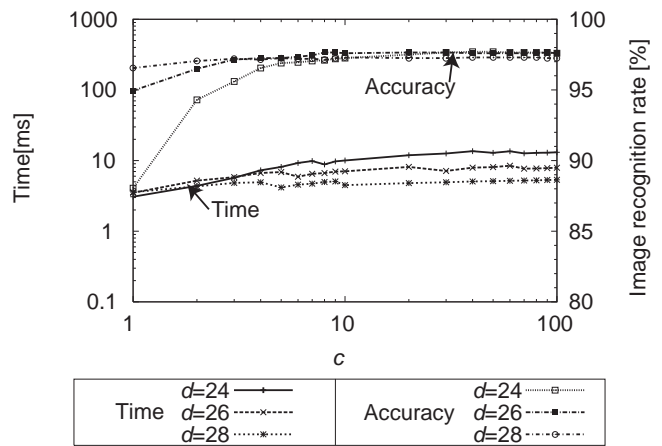
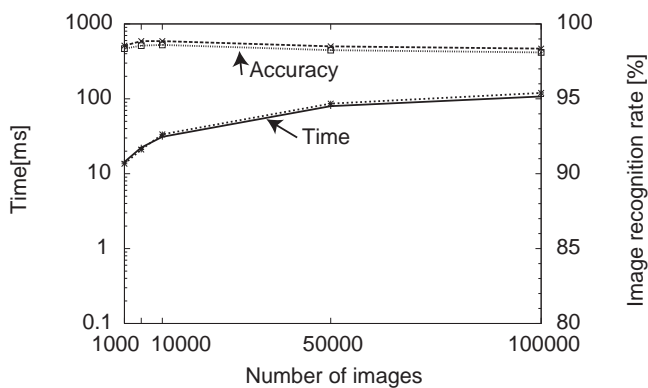


図 8 c と認識率, 処理時間の関係



	16bit	2bit
Time	—●—	—○—
Image recognition rate	—□—	—◇—

図 7 登録画像数と認識率, 処理時間の関係

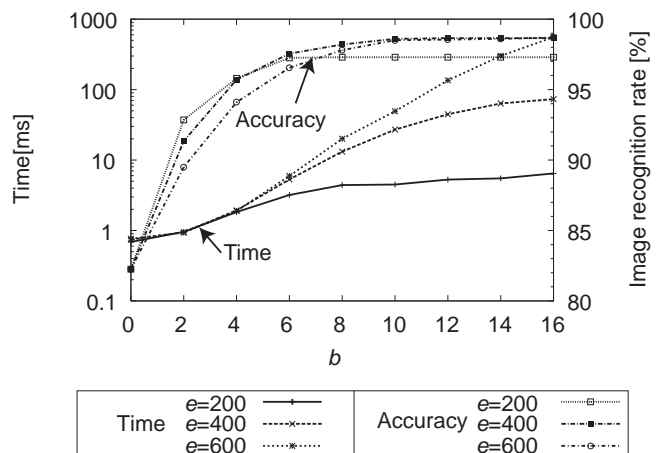


図 9 b, e と認識率, 処理時間の関係

処理時間については, ほとんど変化が見られなかった. 2bit の場合と 16bit の場合の差異について考えてみると, 2bit の場合には検索質問の特徴ベクトルを量子化する処理が余分に必要である. しかし, 検索質問の特徴ベクトルは 600 個程度しかないため, 処理時間は問題とならないと考えられる.

さらに, 提案手法の 2bit と 16bit の場合について誤認識となる検索質問について調べた. パラメータは先ほどと同様に $b = 10, c = 10, d = 28, e = 400$ を用いた. 結果を表 2 に示す. 2bit と 16bit の場合について, 撮影条件ごとに誤認識となった検索質問を \times で示した.

表から, 2bit と 16bit の場合で誤認識の傾向にはほとんど差がないことがわかる. また, 誤認識となる検索質問は特定の登録画像に集中していることがわかった. これらの画像が誤認識を起こしやすい理由は, エッジがはっきりせずぼやけているため, 検索質問の特徴点数が少なくなってしまうためであると考えられる.



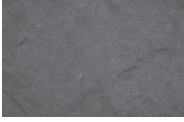



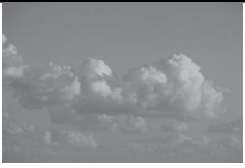





4.4 提案手法のパラメータと認識率・処理時間

ここまでの実験で, 2bit で量子化すれば認識率を落とさずにデータ量を削減できることがわかった. 本節では, 2bit で量子化した場合について提案手法のパラメータと認識率処理時間の関係に大きな変化がみられないかを確認する. 提案手

法の主なパラメータである b, c, d, e について実験を行った. まず, 衝突の閾値 c とハッシュ構築に使う次元数 d と認識率, 処理時間の関係について述べる. このとき画像データベースは 1 万枚, ハッシュ表のサイズとしては $H_{\text{size}} = 2^d$ とした. $e = 200, b = 10, d = 24, 26, 28$ とし, c を変化させた結果を図 8 に示す. c が減少するにつれ, 処理時間が減少していることがわかる. ただし, c を小さくしすぎると, 認識率が低下した. これは, 認識に寄与していたものも削除してしまったためと考えられる. 一方, c を増加させた場合に, 計算時間は増加するものの, 認識率が減少することはほとんどなかった. これは, 最近傍にはなり得ない特徴ベクトルを検索したとしても, 距離計算によって排除可能なためと考えられる.

次に, 両方試す次元数の閾値 b と認識率, 処理時間の関係について述べる. $d = 28$ とした上で, $e = 200, 400, 600, c = 10$ とし, b を変化させた結果を図 9 に示す. b を増加させると, 処理時間は増加するものの, 認識率は改善する. 両方試す処理の対象とする幅 e が小さいものほど, b が小さい場合に認識率がよく上昇することがわかる. これは次のように説明できる. 検索処理では, 検索質問の特徴ベクトルの後ろの次元から $-e$ から e の範囲内のもの b 個を処理の対象とする. このとき, e が大きければ, 処理が必要な部分に到達するまえに b 個の閾値にかかるため, 認識率が低下する. また, ある程度 b を増やすと, b

表 2 誤認識となる検索質問

DB 登録画像	撮影条件と量子化 bit 数								DB 登録画像	撮影条件と量子化 bit 数							
	60°		75°		90°		一部			60°		75°		90°		一部	
	2	16	2	16	2	16	2	16		2	16	2	16	2	16		
	x	x	x	x	x	x				x	x	x	x	x	x	x	x
特徴点数:96	54		65		60		180		特徴点数:15	40		54		50		3	
	x	x	x	x	x	x	x	x		x							
特徴点数:52	57		79		74		44		特徴点数:1525	134		161		161		573	
	x	x	x	x	x					x							
特徴点数:15	76		67		62		10		特徴点数:171	77		97		81		70	
							x			x	x			x		x	x
特徴点数:71	58		69		56		6		特徴点数:90	108		104		109		62	
	x	x	x	x	x	x	x	x		x							
特徴点数:325	70		75		87		25		特徴点数:257	107		110		105		118	
	x	x	x	x	x	x				x							
特徴点数:4246	75		89		98		392		特徴点数:135	79		83		87		163	

の増加に対して認識率，処理時間とも伸びが鈍くなる．これは各次元の値が $-e$ から e の範囲にあるベクトルがそれ以上存在せず，処理の対象とするインデックスが増加しないからである．

これらのことから， e をあまり大きな値にしなければよい結果が得られることがわかる．

以上の実験の結果は，野口らの手法で得られたものと大きな差は見られなかった．そこから，量子化がパラメータに与える影響は軽微であるといえる．

4.5 ANN との比較

近似最近傍探索の従来法である ANN に対して，2bit で量子化を行った場合と比較するため，1 万枚の画像を登録しパラメータをさまざまに変え，横軸に認識率，縦軸に処理時間を描いたグラフを図 10 に示す．

ANN で許容誤差 ϵ を 2 から 1000 まで変化させたものを線

で描き，評価の基準とした．右にプロットされているものほど認識率が高く，下にプロットされているものほど処理時間が短い．そのため，右下にプロットされているものほど優れていると言える．このグラフからわかる通り，提案手法は ANN を用いた場合よりも優れているといえる．

次に，各種パラメータの代表的な値を用いた認識率と処理時間を表 3 に示す．まず，ANN について $\epsilon = 6$ とした場合の認識率と処理時間についてみる．2bit で量子化すると認識率は 98.8% から 97.8% に低下し，処理時間は約 1/4 となった．これは，量子化の影響により距離計算の対象となるベクトルが減少したためと考えられる．そこで，同じ認識率が得られるパラメータで比較すると，2bit では $\epsilon = 3$ となる．この場合，処理時間が大幅に増加することがわかる．これは，量子化誤差により低下した認識率を元の認識率に戻すためには，近似を弱くし

表 3 各手法の認識率 [%] と処理時間 [ms]

手法	パラメータ	60 度		75 度		90 度		一部		平均	
		精度	時間	精度	時間	精度	時間	精度	時間	精度	時間
ANN 16bit	$\epsilon = 3$	98.8	973.7	99.0	1004.0	99.0	1005.5	99.4	3165.4	99.1	1537.2
	$\epsilon = 6$	98.4	147.3	98.8	155.6	98.8	155.3	99.0	468.2	98.8	231.6
	$\epsilon = 20$	96.0	15.9	97.8	18.3	97.8	18.4	98.0	50.8	97.4	25.9
ANN 2bit	$\epsilon = 3$	98.6	562.2	99.0	578.0	98.8	576.2	98.8	1835.6	98.8	888.0
	$\epsilon = 6$	96.6	28.9	98.4	32.5	98.4	32.4	97.6	92.4	97.8	46.5
	$\epsilon = 20$	69.0	2.4	92.4	3.0	93.8	3.1	93.6	8.6	87.2	4.3
提案手法 16bit	$b = 10, c = 10, d = 24, e = 600$	98.8	111.7	99.0	136.2	98.6	139.0	99.0	373.9	98.9	190.2
	$b = 12, c = 10, d = 28, e = 400$	98.2	33.3	98.8	40.6	98.8	41.5	99.2	111.2	98.8	56.6
	$b = 6, c = 10, d = 28, e = 200$	95.0	1.7	98.4	2.2	98.0	2.3	97.8	6.4	97.3	3.1
提案手法 2bit	$b = 10, c = 10, d = 24, e = 600$	98.2	135.9	98.8	165.8	98.8	169.2	98.8	453.8	98.7	231.2
	$b = 12, c = 10, d = 28, e = 400$	98.2	26.3	98.8	32.0	98.6	32.8	99.0	87.6	98.7	44.7
	$b = 6, c = 10, d = 28, e = 200$	95.4	1.7	98.2	2.2	98.0	2.3	97.4	6.5	97.3	3.2

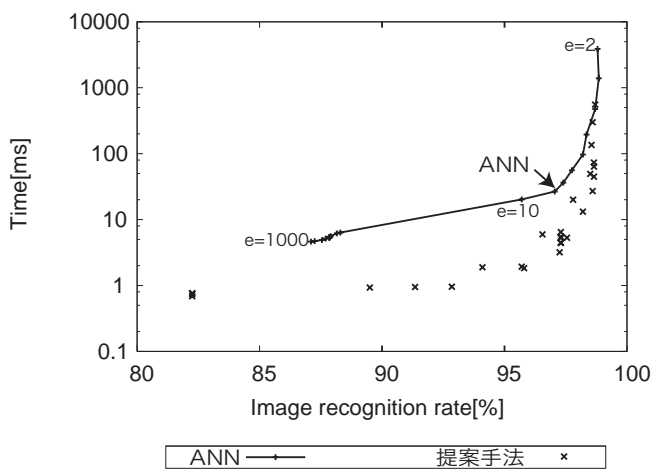


図 10 各手法の比較

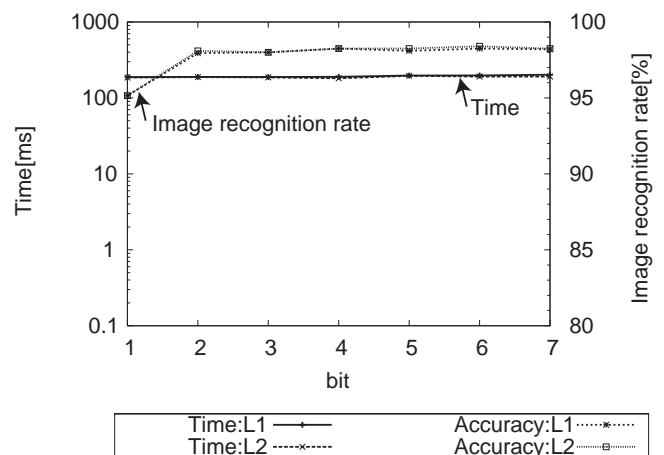


図 11 距離尺度と認識率・処理時間

て、より正確な探索を行う必要があるためと考えられる。

提案手法について見てみると、同じパラメータでは認識率が少し落ちることがわかる。そのため、同じ認識率を得るためには余分に時間が必要である。

以上から、量子化を行うと、同じ認識率を得るために必要な処理時間は増加することがわかる。しかし、それと引き換えに、量子化によってメモリ使用量を大幅に削減できるという利点を得ることができる。

4.6 距離尺度

これまでの実験では、距離尺度としてユークリッド距離 (L2) を用いていた。そこで他の距離尺度を用いた際の影響を調べるため、そこで、マンハッタン距離 (L1) とユークリッド距離 (L2) の比較を行った。

実験条件としては、4.3 と同じものを用い、登録画像数が 10 万枚のときに比較した。結果を図 11 に示す。図より認識率、処理時間ともに変化が見られないことがわかる。

5. まとめ

本稿では、特徴ベクトルをスカラー量子化することで、メモリ使用量を削減する手法を提案した。各次元 2bit で量子化し

た場合に、量子化なしの場合と比べ処理時間を犠牲にすることなく、メモリ使用量を 1/3 にすることができた。また、メモリ使用量、処理時間、認識率はそれぞれトレードオフの関係にあることがわかった。

今後の課題としては、visual word による手法との比較や、さらなる大規模なデータでの実験が挙げられる。

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] D. Lowe, "Distinctive image features from scale-invariant keypoints," International Journal of Computer Vision, vol.60, no.2, pp.91-110, 2004.
- [2] J. Ponce, M. Hebert, C. Schmid and A. Zisserman Eds., Toward Category-Level Object Recognition, Springer, 2006.
- [3] J. Sivic and A. Zisserman, Video google: A text retrieval approach to object matching in videos, Proc. ICCV2003, Vol. 2, pp.1470-1477, 2003.
- [4] D. Nistér and H. Stewénus, Scalable recognition with a vocabulary tree, Proc. CVPR2006, pp.775-781, 2006.
- [5] 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, pp.111-118, July, 2007.
- [6] S. Arya, D. M. Mount, R. Silverman and A. Y. Wu, "An optimal algorithm for approximate nearest neighbor searching," Journal of the ACM, vol.45, no.6, pp.891-923, 1998.