

## 特定物体認識

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: †kise@cs.osakafu-u.ac.jp

**あらまし** 特定物体認識とは、画像内にある個別物体（物体のインスタンス）を認識する処理である。換言すれば、物体の「見え」から物体 ID への変換を意味する。これが実現できれば、物体を Web のリンクアンカーとして用いることができ、Web を実世界に拡張することが可能となる。我々の周囲には無数の物体インスタンスがあるため、特定物体認識の実現は、規模との戦いであると言える。本稿では、局所特徴量を用いた特定物体認識を中心に、最近の動向について述べる。

**キーワード** 特定物体認識, 局所特徴量, Bag-of-Features, 近似最近傍探索

## Specific Object Recognition

Koichi KISE†

† Graduate School of Engineering, Osaka Prefecture University

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

E-mail: †kise@cs.osakafu-u.ac.jp

**Abstract** Specific object recognition is a process to recognize instances of objects captured as images. In other words, it indicates the conversion from an appearance of an object to its ID. This functionality allows us to use objects as link anchors of the web, and thus extend the cyber world to the real world. Since we are surrounded by an immense number of object instances, specific object recognition, in its nature, is a challenge to the scalability. In this tutorial, I show recent advances in specific object recognition with the special focus on methods using local features.

**Key words** Specific object recognition, Local feature, Bag-of-Features, Approximate nearest neighbor search

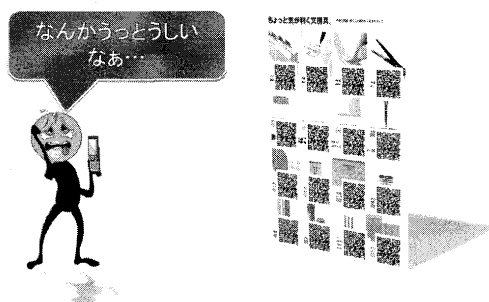
### 1. ま え が き

物体認識とは、画像中の物体が何であるのかを言い当てる処理である。そのタスクは、大きく一般物体認識 (generic object recognition) [1] と特定物体認識 (specific object recognition, particular object recognition) [2] に分類できる。一般物体認識とは、画像に写っている物体のクラスを認識する処理である。一方、特定物体認識とは、画像中の個別物体（物体のインスタンス）を認識する処理である。前者が一般的な「車」というカテゴリを回答する処理であるのに対して、後者は車の特定のモデル名を回答する処理と言える。本稿は、後者に焦点を当てて述べる。なお、特定物体認識は、物体検索、画像検索などと呼ばれることもある。また、類似画像検出 (near duplicate detection) [3], [4] とも関連が深い。

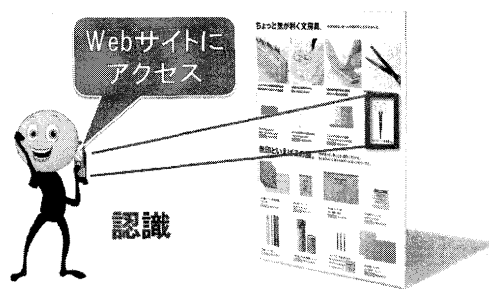
特定物体認識は、物体の「見え」を入力とし、その物体の ID を回答する処理と見ることができる。では、見えを ID に変換できると、どのようなことが可能になるだろうか。まず、写真の自動索引付けが考えられる。例えば、観光地で撮影した写真

に著名な建物が写っているとき、その建物の名前を索引として付けることができる。また、物体そのものをバーコードの代わりに用いることができる。特定物体認識を用いると、物体を撮影することにより、その物体に関連付けられたサービスにアクセス可能となる。図 1(a) の例のように、バーコードはしばしば物体の外観を損ねるが、図 1(b) のように特定物体認識を用いるとその心配もない。上記は情報を取り出す処理であるが、特定物体認識が可能であれば、逆に物体に情報を関連付けることも容易である。これは、物体の画像と関連情報をデータベースに新たに格納すれば実現できる。バーコードを用いる場合には、物体にバーコードを貼らねばならないが、美術品などの例を考えると、これは常に可能とは限らない。また、ポスターのように同一の物体が多数存在する場合、全ての物体にバーコードを貼り付けることは現実的ではない。物体認識を用いると画像を撮影するだけで登録が可能であるため、このような問題は生じない。

特定物体認識を実現する方法には様々なものが考えられるが、本稿では特に局所特徴量 (local feature) に基づく手法に焦点を当



(a) Barcode



(b) Specific object recognition

図1 バーコード vs. 特定物体認識.

てる。これは現在の研究の大半がこの範疇に属することによる。先駆的研究としては、Schmid ら [5] と Lowe [6] を挙げることができる。特に後者で提案された SIFT (Scale-Invariant Feature Transform) は、標準的な局所特徴量としての地位を築いている。

SIFT などの局所特徴量を用いて特定物体認識を実現する最も単純な方法は、次の投票方式である。まず、認識対象となる物体の画像（以下、DB 画像と呼ぶ）から局所特徴量を抽出し、物体の ID と関連付けて予めデータベースに収めておく。通常、画像 1 枚あたり数百から数千、場合によっては数万の局所特徴量が得られる。次に、認識したい物体が写った画像（以下、検索質問画像と呼ぶ）からも局所特徴量を抽出し、データベースの局所特徴量と照合する。このとき、局所特徴量の照合結果を物体に対する投票と考え、最終的に得票数が最大となる物体を認識結果とする。実際、SIFT 特徴量を用いてこの単純なプロセスを実現し、特定物体認識システムを作ると、驚くほど高い認識率を得ることが可能である [2]。

では、特定物体認識を考える上で何が問題となるであろうか。最大の問題として本稿で注目するのは「規模」である。一般物体認識の場合、認識対象（物体のクラス）の数は、およそ  $10^2$  から  $10^4$  規模である。一口でいえばキロオーダーの対象数である。一方、特定物体認識の研究では、認識対象（物体のインスタンス）の数は、 $10^6$  から  $10^9$  規模（メガからギガオーダー）になりつつある。

規模の問題を具体的に捉えるため、上記の単純な方法を 1M 枚の画像の大規模データに適用することを考えてみよう。まず、メモリ量を取り上げる。DB 画像 1 枚から得られる局所特

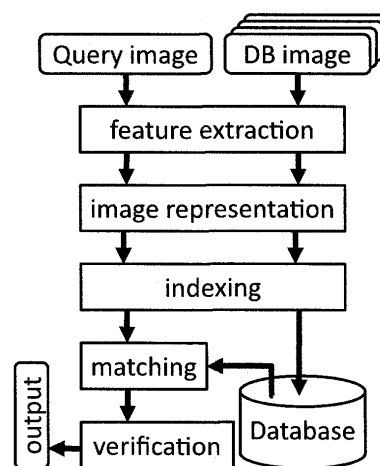


図2 特定物体認識処理の流れ.

微量の数を 1K 個とすると、1M 枚の画像データベースから得られる局所特徴量の総数は 1G 個となる。これは、局所特徴量の次元数を 128 次元とし、各次元 2Byte で表現する場合、総容量 256GB となる量である。効率的な処理のためには局所特徴量が主記憶に収められている必要があるが、この量でそれを満たすのは簡単ではない。仮に可能であっても、この千倍である 1G 枚までの大規模化は望みが薄い。処理時間については次の通りである。検索質問画像からも同数の局所特徴量が得られるとき、単純に全数照合を行うとその回数は 1T(テラ) 回となる。これは、最近の計算機を使っても、終了までに 3 日を要する処理であり、ユーザが待てる時間を大幅に超えている。

以上から、メモリ量と処理時間の問題を解決することが、特定物体認識の実用化にとって本質的であることが分かる。後に示すようにメモリ量や処理時間の削減は、認識率の低下をもたらすため、これらの間のバランスをいかに確保するかが研究者の苦心のしどころと言える。本稿では、1M 枚あるいはそれを越える画像を対象とした特定物体認識を、実際的な時間(数十ミリ秒から数秒)とメモリ量(数百 MB~数十 GB)で行うための、これまでの試みをまとめる。

## 2. 処理の流れ

特定物体認識処理の流れを図 2 に示す。先に、右側の矢印で示すデータベース側の処理から述べる。まず、特徴抽出 (feature extraction) の処理により、DB 画像 (DB image) から局所特徴量を抽出し、画像表現 (image representation) を得る。これは通常高次元ベクトルである。次に、後の照合を高速化するため、索引付け (indexing) を行った上で画像表現をデータベースに格納する。一方、左側に示す検索質問側の処理は次の通りである。検索質問画像についても、特徴抽出、画像表現、索引付けの処理を順に施す。そして、データベース中の画像表現と照合 (matching) する。最後に検証 (verification) を適用し、結果を出力する。上記の処理は、特定物体認識の目的によって変化する。また一部、適用されないものもある<sup>(注1)</sup>。以下では、そのような場合も含めて順に述べていく。

(注1) : 1. で述べた投票による特定物体認識では検証を行っていない。

### 3. 特徴抽出

一般に、どのような特徴を抽出すればよいのかは、DB 画像と検索質問画像がどの程度異なるかによる。Near-duplicate と呼ばれるほぼ同じ画像に対象を限ることができるなら、画像全体から得られる色ヒストグラムなどの大域特徴量 (global feature) を用いることも可能である [7]。ところが、物体を撮影する角度やその際の照明条件などの撮影条件が大きく変動する場合、色ヒストグラムなどの単純な特徴量で認識することは困難である。また、撮影される物体の部分が、DB 画像と検索質問画像で異なる場合、大域特徴量の値が大きく変動するため、同様に認識が困難になる。

これらの問題に対処する方法として、局所特徴量が近年注目を集めている。局所特徴量を抽出する処理は、大きく、検出 (detection) と記述 (description) に分割できる。まず、検出処理により、画像から特徴的な局所領域を取り出す。画像にも依るが、一般に数百から数千の局所領域が得られる。次に、得られた局所領域から、記述処理により特徴ベクトルを抽出する。

特徴抽出は、再現性 (repeatability) <sup>(注2)</sup> と識別性 (discriminative power) という 2 つの指標で評価される。再現性とは、撮影条件の変動が加わっても、同じ物体の同じ部分からは同じ特徴量が得られるかどうか、識別性とは、異なる物体から得られた特徴量が互いに十分異なるかどうかである。

再現性は検出と記述の両面に関わる。高い再現性を持つ検出処理は、変動の下でも同じ局所領域を安定的に取り出すことができる。例えば、SIFT は相似変換 (similarity transform) というクラスの幾何変換に対して頑健な検出処理となっている。また、Hessian-Affine, Harris-Affine, MSER などの検出処理を用いると、相似変換よりも広いアフィン変換 (affine transform) に対しても安定した領域を得ることができる。さらに、記述処理が高い再現性を持つと、例えば、画像中での物体の移動や拡大縮小、照明の変動があっても、類似性の高い特徴ベクトルを取り出すことが可能である。

識別性は物体を他のものと区別する上で重要な性質である。再現性が高くても、異なる物体から類似の特徴ベクトルがしばしば得られるような記述処理では、物体を正しく認識することはできない。通常、多くの記述処理では、高い識別性を得るために、高次元の特徴ベクトルを抽出する。SIFT の例では、128 次元のベクトルが抽出される。ただし、次元数が高いことは良いことばかりではない。メモリ量の観点からは、一般に次元数が低い方がよい。

他に、検出処理の重要な性質として密度がある。密度とは、一定の面積からどの程度の数の局所領域が取り出されるかを表す指標である。一般に、密度が高いと認識に有利である [8]。残念ながら、より広範囲の幾何変換に不変な検出処理は、密度が低いという傾向にある。

これまでに極めて多数の検出および記述の処理が提案されており、また、その比較も行われている [9], [10]。得意・不得意

はあるものの、一般的に性能が高いと言われているのは、SIFT (検出と記述の両方) である。また、その拡張である PCA-SIFT は、再現性や識別性をほぼ保ったまま低次元の特徴ベクトルが得られる (文献 [11] では 36 次元)。

### 4. 画像表現

特徴抽出が終了すると、次のステップは画像表現となる。まえがきで述べたように、最も単純な方法は、局所特徴量をすべて用いて画像を表現する方法であるが、メモリ量と処理時間の問題が生じる。ここでは、このうちメモリ量の問題をいかに解決するかについて、これまでの試みをまとめる。

#### 4.1 ベクトル量子化と BOF

メモリ量の問題を解決する一つの方法は、量子化によって局所特徴量の記憶に必要なメモリ量を減らすことである。Sivic らはベクトル量子化を用いる手法を提案している [12]。この手法のポイントは、ベクトル量子化によって得られた特徴ベクトルを文書の単語になぞらえて visual word と呼び、Bag-Of-Words (BOW) と呼ばれるテキスト検索のモデルを導入したことである <sup>(注3)</sup>。BOW モデルは、単語の出現順序を捨象し、出現頻度によってテキストを近似表現するものであり、テキスト検索の標準的な手法となっている [13]~[15]。画像の場合のモデルは、Bag-Of-Features (BOF) あるいは Bag-Of-Visual-Words (BOVW) と呼ばれることもある。本稿では以後 BOF と呼ぶ。

BOF に基づく画像表現について考える。いま、十分な数の画像の集合を考え、それらから得られた特徴ベクトルの集合  $P = \{p_i\}$  をベクトル量子化した結果、 $V = \{v_i\}$  ( $1 \leq i \leq m$ ) なる visual word が得られたとする。ここで  $m$  は visual word の数である。また、 $d(\cdot, \cdot)$  を距離とすると、元の特徴ベクトル  $p$  と、それに対応する visual word  $v_i$  には、

$$v_i = \min_{v \in V} d(p, v) \quad (1)$$

という関係がある。すなわち、 $V$  の visual word のうち、 $v_i$  は  $p$  の最近傍となるものである。具体的な距離としては、ユークリッド距離がよく用いられる。

このような、特徴ベクトル  $p$  から visual word  $v_i$  への変換を  $P$  の要素すべてに対して適用すると、画像における visual word の出現が明らかになる。その結果に基づき、画像  $j$  を visual word に関する  $m$  次元ベクトル

$$w_j = (w_{1j}, \dots, w_{mj})^T \quad (2)$$

で表現する。本稿ではこれを BOF 表現と呼ぶ。ここで、 $w_{ij}$  は画像  $j$  における visual word  $v_i$  の重みを表す。具体的には、テキスト検索でよく用いられる TF-IDF (term frequency - inverse document frequency) 重みを利用することが多い。画像  $j$  における visual word  $v_i$  の出現回数を  $tf_{ij}$ 、 $v_i$  が出現した画像数を  $df_i$ 、全画像数を  $N$  とするとき、visual word  $v_i$  の画像  $j$  における重

(注3)：特定物体認識をテキスト検索と関連付けることにより、テキスト検索の分野で開発された、大規模なデータを効率的に扱う様々な手法を「輸入」することが可能となる。この点も重要な貢献であると言える。

(注2)：安定性 (stability) も類似の概念である。

み  $w_{ij}$  は、例えば、

$$w_{ij} = \frac{tf_{ij}}{\sum_i tf_{ij}} \log \frac{N}{df_j} \quad (3)$$

で表される。

以上の BOF 表現は、特定物体認識のみならず一般物体認識にも用いられるものであり、局所特徴量に基づく物体認識のデファクト・スタンダードと言える。ただし、特定物体認識に用いる上では、問題となる事項がいくつかある。

一つは、特徴ベクトルの数である。前述のようにメガからギガオーダーの特徴ベクトルをベクトル量子化することは、かなりの困難を伴う。クラスタリング手法として良く用いられる  $k$ -means アルゴリズムを用いる場合、ある程度、データをサンプリングした上でであっても、計算コストは膨大となる。この問題を解決するための方法として、階層的  $k$ -means アルゴリズム [16] がある。これは、小さな  $k$  に対する  $k$ -means アルゴリズムを再帰的に適用するものである。また、後に述べる近似最近傍探索を用いてクラスタリングの際の距離計算を高速化した approximate  $k$ -means 法もある [17]。

もう一つは、visual word の数  $m$  に関する。  $m$  は、BOF 表現の識別性を左右するパラメータである。一般物体認識の場合、画像の表現は、同じクラスに属する個々の物体の差を吸収することが求められる。このため、visual word の数は元の特徴ベクトルの数に比べて十分小さく設定され、大幅なメモリの節約が可能となる。一方、特定物体認識の場合、個別物体の差を失うことは許されないため、  $m$  は十分大きく設定される。対象や規模にもよるが、  $m$  としては 0.1M~10M 程度を用いることが多い。また、文献 [16] では、元の特徴ベクトル数個に対して 1 つの visual word を設定すると良い結果が得られると述べられている。以上から、visual word の記録にはある程度のメモリが必要となる。また、検索質問画像から得られた特徴ベクトルを visual word に変換するための処理時間が無視できなくなる。したがって、最近の手法では  $m$  を大きくするのではなく、  $m$  を小さく保った上で識別性を他の手法で補うことが多い。この手法については後に述べる。

## 4.2 スカラ量子化

特定物体認識において、識別性の高い BOF 表現を得ることが困難であるのは、visual word の記録にかかるメモリ量と、その照合に必要な処理時間による。後者の問題を回避する一つの方法は、大規模な照合が不要な量子化を行うことである。これはスカラー量子化によって実現できる。

特徴ベクトルを  $\mathbf{p} = (p_1, \dots, p_D)$  とするとき、スカラ量子化では、これを量子化したベクトル  $\mathbf{p}' = (p'_1, \dots, p'_D)$  に変換する。  $k$  レベルの量子化ならば、  $p'_i \in \{1, \dots, k\}$  である。このとき、量子化した値  $1, \dots, k$  の出現確率が等しくなるように閾値を決める。閾値は、次元ごとに  $k-1$  個決めればよい。特徴ベクトルの量子化は、すべての次元において、この閾値と値  $p_i$  の比較により行われる。

それでは、特定物体認識には、どの程度の量子化が必要であろうか。 [18] では、36 次元の PCA-SIFT に対して、次元あたり 2bit でも殆ど有効性を失わないことが示されている。これは、

元の表現が次元あたり 16bit であるならば、1/8 の圧縮が可能であることを意味する。

次元あたりのメモリ量を同一にした場合、スカラ量子化はベクトル量子化に比べて一般に誤差が大きくなる。一方、膨大な数の visual word を別途記録する必要がなく、閾値のみの記録でよいから、メモリ量の面で有利になるとともに、量子化の計算コストが少なく済む。

スカラ量子化を用いた場合の画像表現としては、ベクトル量子化の場合と同様、式 (2) の BOF 表現が考えられる。このとき、BOF 表現の次元数  $m$  は  $m = k^D$  となる。上記の  $k = 4$ 、 $D = 36$  の場合、これは  $10^{21}$  を越える極めて大きな数となるが、後に述べる転置インデックスの利用によって問題とならない。

## 4.3 中間的な量子化

ベクトル量子化は誤差の少ない表現が可能であるものの visual word の記録や照合という問題が生じる。一方、スカラ量子化はベクトル量子化に比べて誤差は大きくなるものの、閾値の記録に必要なメモリ量は少なく、照合の問題も生じない。これらの両極端な方法に対して中間的な方法があると、バランスを取る上で有利である。

その目的を満たす方法として、ここではプロダクト量子化 (product quantization) [19] と Hamming Embedding [20] と呼ばれる方法を紹介する。前者は、スカラ量子化を基本として部分的にベクトル量子化を導入した手法、一方、後者は、逆にベクトル量子化を基本とし、スカラ量子化で補強を行う手法と捉えることができる。

### 4.3.1 プロダクト量子化

プロダクト量子化では、  $D$  次元特徴ベクトル

$$\mathbf{p} = (p_1, \dots, p_D) \quad (4)$$

に対して、  $D' = D/R$  次元の部分ベクトル  $\mathbf{u}_i$  ( $1 \leq i \leq R$ )

$$\mathbf{u}_i = (p_s, \dots, p_e) \quad (5)$$

$$s = D'(i-1) + 1 \quad (6)$$

$$e = D'i \quad (7)$$

を考え、その各々についてベクトル量子化を施す。そして、結果として得られるコードワードを  $R$  個組み合わせることによって、visual word を表現する。部分ベクトル  $\mathbf{u}_i$  から得られるベクトル量子化後のコードワードの数を  $k^*$  とする。そのとき、記憶すべきコードワードの総数は  $Rk^*$  であるものの、全体で表現できる visual word は  $(k^*)^R$  個となる。プロダクト量子化は、  $R = 1$  のときベクトル量子化を、  $R = D$  のときスカラ量子化を表す。

### 4.3.2 Hamming Embedding

ベクトル量子化で visual word を記録するためのメモリ量を削減する一つの方法は visual word の数を減らすことであるが、それにより識別性を失う点が問題となる。これを部分的なスカラ量子化の導入によって補う手法が Hamming Embedding である。

図 3 に具体例を示す。ここで、黒点は visual word を表し、線は特徴空間での境界線を表す。線によって区切られた領域はボ

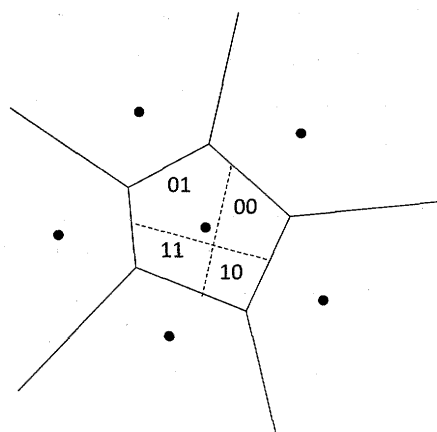


図3 Hamming Embedding.

ロノイ領域である。Hamming Embedding では、ベクトル量子化によって失われた識別性を回復するため、ボロノイ領域をスカラ量子化によって分割する。具体的な手順は以下の通りである。まず、元の  $D$  次元特徴ベクトル  $\mathbf{p} = (p_1, \dots, p_D)$  を次元削減し、 $D_b$  次元とする。次に、各次元に閾値処理を施し、ビットベクトル  $b(\mathbf{p}) = (b_1(\mathbf{p}), \dots, b_{D_b}(\mathbf{p}))$  に変換する。図3の“00”、“01”などは、このビットベクトルの例である。

Hamming Embedding を用いた2つの特徴ベクトル  $\mathbf{p}$  と  $\mathbf{q}$  の照合は、次のように行う。通常のベクトル量子化の場合、2つの特徴ベクトルが同じ visual word  $\mathbf{v}_i$  に対応付くならば、 $\mathbf{v}_i$  に対して定められた TF-IDF 重み  $w_{ij}$  (式(3)) が得られる。一方、HE 表現を用いる場合は、同じ visual word に対応付くだけでなく、両者のビットベクトルに対するハミング距離  $h(b(\mathbf{p}), b(\mathbf{q}))$  が一定の閾値以下でなければならないという条件が付く。これにより、同じ visual word に対応付く場合でも、両者が十分類似していなければ照合されないため、識別性が増す。

#### 4.4 Soft Assignment

最後に紹介する手法は、visual word に何も加えることなく、識別性を増す手法である。

ベクトル量子化では、特徴ベクトルから visual word への対応付けは、ユークリッド距離を用いて一意に定まる。このようなハードな割り当てを用いると、特徴ベクトルが visual word で定まるボロノイ領域の境界線近くにある場合、問題が生じる。すなわち、画像の変動によって特徴ベクトルの値が変化すると、別の visual word に対応付く可能性が出てくる。この問題に対処する一つの方法は、ハードではなくソフトな割り当てを行うことである。[21] では、特徴ベクトルの  $k$ -NN となる  $k$  個の visual word について、距離に応じた重みをつけて割り当てを行っている。これにより、従来法であれば同一の visual word に割り当てられる特徴ベクトルが、ボロノイ領域内の位置によって、異なる値を持つことができるため、識別性が増す。また、変動によって境界線を越えてしまう場合であっても、大きな値の変動を受けずに済む。なお、 $k$  の値として、[21] では3を主に用いている。

ただし、この手法には次の注意が必要である。1つの特徴ベクトルに対して  $k$  倍の visual word が割り当てられるというこ

とは、その分、BOF 表現 (式(2)) のスパース性が失われることになる。これは、次節で述べる転置インデックスを用いる場合、より大きなメモリ量が必要なことを意味する。

## 5. 索引付けと照合

次の2ステップは索引付けとそれに基づく照合である。説明の都合上、これらを併せて述べる。

### 5.1 転置インデックスと照合

前節の議論によって、DB 画像と検索質問画像は共に式(2)の BOF 表現によって記述されている。いま、この表現をそれぞれ  $\mathbf{w}_j$ ,  $\mathbf{w}_q$  とする。認識のためにはこれらを照合し、最も類似するもの、あるいは最も類似するものから上位いくつかを取り出す必要がある。このための類似度 (あるいは相違度) としては、様々なものが用いられる。例えば、 $L_1$  距離、ユークリッド距離、ベクトルをヒストグラムとして見たときのヒストグラムインタセクション (histogram intersection), あるいは、情報検索で一般的なコサイン尺度

$$\text{sim}(\mathbf{w}_i, \mathbf{w}_q) = \frac{\mathbf{w}_i \cdot \mathbf{w}_q}{\|\mathbf{w}_i\| \|\mathbf{w}_q\|} \quad (8)$$

などがある<sup>(注4)</sup>。ところが、これらを直接計算することは、次の理由により現実的ではない。

- 十分な識別性を得るには膨大な数の visual word が必要となる。そのため、式(2)の BOF 表現は高次元となり、メモリ量や処理時間への負担が大きい。
- 式(2)のベクトルは DB 画像の数だけ存在する。そのため、データベースが大規模であればメモリ量や処理時間への負担が大きい。

この問題は、転置インデックス (inverted index) という技術を用いると解決できる。この技術のポイントは、BOF 表現が極めてスパース (ほとんどすべての要素が0) なことを利用する点にある。以下では、転置インデックスの基本的な事項のみを述べる。効率的な照合方法、インデックスの圧縮など、より詳しい事項に興味のある読者は、[13], [14] を参照されたい。

図4に転置インデックスの簡単な例を示す。図4左に示すように、いま visual word として  $v_1, \dots, v_6$  があり、 $\mathbf{w}_1$  と  $\mathbf{w}_2$  の2つの BOF で表された画像がデータベースに収められているとする。これに対して、転置インデックスは、visual word からそれが存在する画像 (とその頻度) に効率的にアクセスするための索引であり、図4右に示すリスト構造となる。認識対象の画像から得たベクトルを  $\mathbf{q}$  とすると、 $\mathbf{q}$  のゼロでない要素について転置インデックスにアクセスし、その visual word が存在する画像とその重みを知ることができる。また、前述のコサイン尺度を計算する際に必要となる内積は、ゼロでない要素の重みを掛け合わせれば得られる。図4の例では、 $(\mathbf{w}_1, \mathbf{q}) = 2 \cdot 1$ 、 $(\mathbf{w}_2, \mathbf{q}) = 3 \cdot 1 + 1 \cdot 1$  となる。特定物体認識では、前述のよう

(注4) : DB 画像のベクトル  $\mathbf{w}_i$  と検索質問画像のベクトル  $\mathbf{w}_q$  を共に正規化しておけば、コサイン尺度は内積によって計算できる。また、これにより、ユークリッド距離とは  $d(\mathbf{w}_i, \mathbf{w}_q) = 2(1 - \text{sim}(\mathbf{w}_i, \mathbf{w}_q))$  という関係が成り立つので、得られる結果は変わらない。

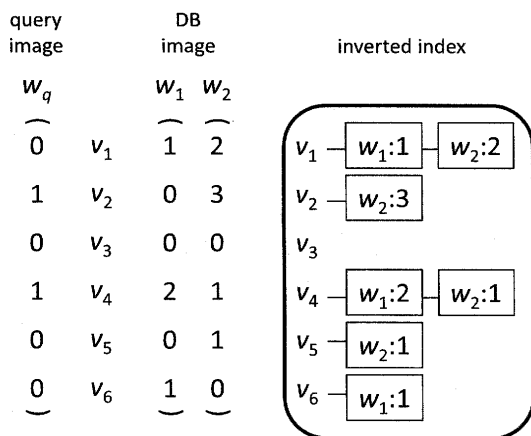


図4 転置インデックス.

に BOF 表現は一般に極めてスパースであるため、ゼロ以外の部分だけを計算対象にすると、内積計算の高速化が可能となる。

## 5.2 投票方式との等価性

最も単純な特定物体認識の手法として、まきがきでは局所特徴量の照合と投票による認識手法を導入した。話が少し横道にそれるが、ここでは、この投票方式による認識が、BOF モデルによる認識の極端な例として捉えられることを述べる [20]。

簡単のため、DB 画像から抽出された特徴ベクトルはすべて異なるものとする。まず、これらの特徴ベクトルをすべて visual word と考える。いま、画像  $j$  から抽出された visual word を  $v_{x+1}, \dots, v_{x+J}$  の  $J$  個とすると、画像  $j$  の BOF 表現は、

$$w_j = (\underbrace{0, \dots, 0}_{x \text{ 個}}, \underbrace{1, \dots, 1}_{J \text{ 個}}) \quad (9)$$

となる。一方、クエリから得た特徴ベクトルの集合  $\{q\}$  の要素をすべて最近傍探索によって visual word のいずれかに対応付け、同様に BOF 表現  $w_q = (q_1, \dots, q_i, \dots)$  を得る。ここで、 $q_i$  は、特徴ベクトル  $q$  のいずれかが visual word  $v_i$  に対応付いたときに 1、それ以外の場合に 0 となる。画像  $j$  との類似度を内積によって定義すると、その値は、

$$\sum_{i=x+1}^{x+J} q_i \quad (10)$$

である。これは、局所特徴量の照合と投票によって得られる画像  $j$  の得票数に他ならない。言うまでもなく最大得票数となる画像は、類似度が最大のものとなる。

なお、上記の特別な場合だけではなく、一般の BOF モデルの場合についても、同様に投票として解釈することが可能である [20]。

## 5.3 visual word との効率的な照合

話を転置インデックスに戻そう。前述のように、転置インデックスを用いると、効率的に類似度を計算することが可能となる。残る問題は、検索質問画像の BOF 表現を、どのように効率的に計算するかである。前述のように、特定物体認識では一般に visual word の数が膨大となるため、式 (1) で定義される正確な最近傍を計算することは困難となる。この問題に対処する一つ

の方法は、正確な最近傍を諦め、近似で済ませることである。このような手法は近似最近傍探索 (approximate nearest neighbor search) と呼ばれ、活発に研究されていると共に、物体認識にも精力的に用いられている [22]。

### 5.3.1 Vocabulary Tree

近似最近傍を求める素朴な方法は、visual word を求めるときに階層的  $k$ -means 法によって得た木構造を利用することである。Nistér らはこれを vocabulary tree と呼んでいる [16]。木の各ノードはクラスタリングによって得られた重心ベクトルに対応し、特に葉は visual word を表す。検索質問画像から得られた特徴ベクトル  $q$  を visual word に対応付けるには、根から順番に葉に向かって木を辿っていけばよい。あるノードまで辿っている状態で、子ノードを選択する方法は以下の通りである。特徴ベクトル  $q$  と子ノードが表す重心ベクトルを比較し、最も距離の小さいものを選択する。ただし、この方法で、式 (1) を満たす結果が得られるとは限らない点に注意が必要である。

### 5.3.2 一般的な近似最近傍探索による照合

物体認識とは独立に提案されている一般的な近似最近傍探索を用いることも可能である。代表的な手法としては、木構造 (kd-tree) をベースにした ANN [23]、ハッシュをベースにした LSH (Locality-Sensitive Hashing) [24], [25]、PCH (Principal Component Hashing) [26]、min-Hash [4]、Semantic Hashing [27]、Spectral Hashing [28] などがある。Lowe による先駆的な研究 [6] でも Best-Bin-First algorithm と呼ばれる木構造ベースの近似最近傍探索が用いられている。

以下では、Lowe の手法とも関連が深く、ソフトウェア [29] が利用可能なため容易に試すことのできる手法として、ANN による近似最近傍探索を紹介する。ANN を用いた特定物体認識の実現については [2] にも記述がある。

図 5 に概要を示す。ここで①などの数字はベクトル (この場合は visual word のベクトル) に対応し、図 5(a) は特徴空間の構造を、(b) はその kd-tree 表現を示す。また、木構造の葉以外のノードは特徴空間の分割を表し、葉ノードはベクトルに対応する。空間分割の仕方は  $k$ -means 法と異なっている。具体的には、空間の軸を一つ選び、閾値を設定して 2 分割する。軸の選び方や閾値の設定方法には様々なものがある [23]。kd-tree にベクトルの集合を収めるには、領域中のベクトルが唯一になるまで、上記の空間分割を再帰的に繰り返せばよい。分割の結果、最終的に得られる領域をセルと呼ぶ。

kd-tree を用いた近似最近傍探索は図 5 右上に記載の手順で実行される。まず、最近傍を求める対象のベクトル (検索質問: query) と根ノードの閾値を比較し、検索質問が特徴空間のどちら側に存在するのかを判定する。この処理を再帰的に繰り返して木を降りていき、最終的に葉ノードに至る。これにより、検索質問が属するセルを発見できる。

セルにはベクトルが一つ対応付いている。それとの距離を  $d$  とすると、真の最近傍は  $d$  を半径とする超球内に存在する。したがって、この超球と重なりを持つセルをすべて調べ、距離が最小となるものを発見すれば最近傍が得られる。図 5(a) の場合は、重なりを持つセルは①、②、④、⑥であり、③と⑤のセル

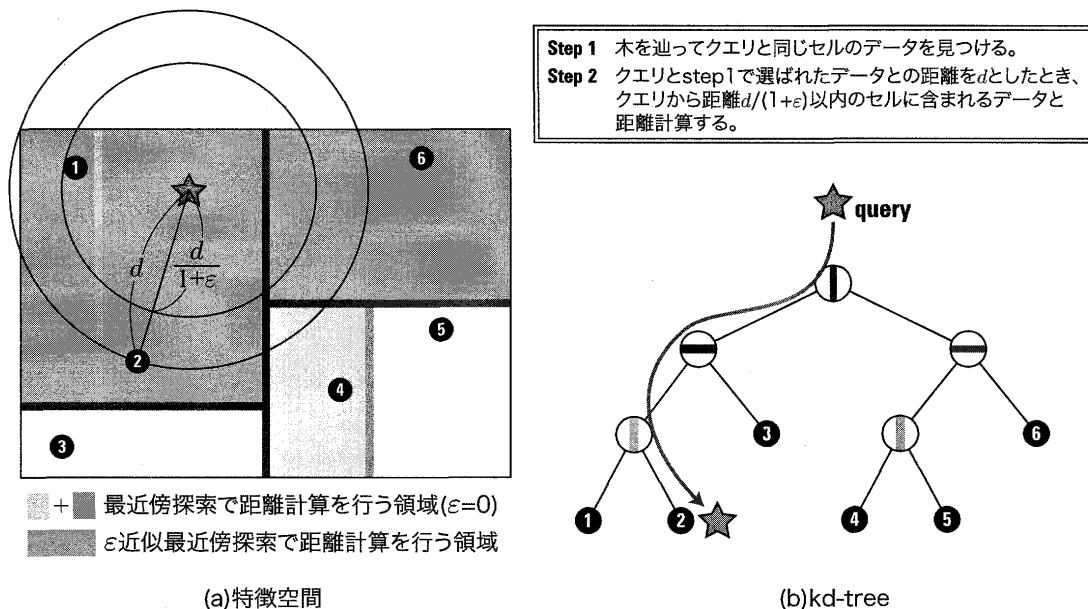


図5 ANNによる近似最近傍探索。

を調べなくても良いことがわかる。ただし、ベクトルが高次元(おおよそ15次元以上)になると、球面集中現象によって超球が大半のセルと重なりを持つようになり、前述のような削減効果を得ることができない。そこでANNでは、超球の半径を $1/(1+\epsilon)$ 倍( $\epsilon > 0$ )することによって縮小し、検査の対象となるセルを減らす。同図の例では、④をさらに削減している。高次元の場合、ベクトルが球面に集中しているため、 $\epsilon$ の値を大きめ(例えば10程度)にとると、大半のベクトルが超球外に追い出される。これにより、正しい最近傍が得られなくなるリスクは負うものの、大幅な高速化が可能となる。

### 5.3.3 ハッシュを用いた照合

木構造ではなくハッシュを用いても効率的な照合を実現できる。ハッシュを用いる近似最近傍探索の代表格はLSHであるが、大規模データには必ずしも適さない。ここでは、単純ではあるが高速な野口らの手法[18]を紹介する。

この手法では、局所特徴量を表す特徴ベクトルをビットベクトルに変換し、そのビットベクトルが表すビンに元の特徴ベクトルをチェーン法で格納する。検索質問についても同様にビットベクトルに変換した後、対応するビンにアクセスし、格納された特徴ベクトルのリストを取り出す。そして、検索質問の特徴ベクトルとの距離を計算し、最近傍となるものを結果とする。リストには必ずしも真の最近傍が含まれているとは限らないため、近似最近傍探索となっている。なお、この手法では、真の最近傍が得られる可能性を高めるため、検索質問のビットベクトルとして複数通りを作成し、ビンにアクセスしている。ビットベクトルを多数生成するほど、処理時間は増すものの、近似の程度が小幅になり、探索は正確になる。

## 5.4 その他の工夫

以下では、索引付けと照合に関する最近の成果の概要を紹介する。

### 5.4.1 高速化

近似最近傍探索を用いて照合を行う場合、どの程度の近似を

すればよいのかが問題となる。例えば、図5に示したANNの $\epsilon$ 、5.3.3で述べたビットベクトルの数、などの近似パラメータの設定である。近似の程度が大幅になれば(先の例では、 $\epsilon$ が小さい、ビットベクトルの数が少ない場合)、それだけ処理時間は短縮できるものの、照合の精度は低下するため、認識率は低くなる。一方、近似の程度が小幅になれば逆の状況が生じる。問題は、認識対象の画像によって、適切な近似の程度が異なる点である。多くの画像に対して有効な近似の程度を設定しようとすると、安全側、つまりあまり近似を行わないような値に、近似パラメータを設定しなければならず、結果として処理時間が長くなってしまふ。

この問題は近似の程度を認識対象に応じて適応的に変化させると解決できる。[30]では、そのような方法として、近似最近傍探索の多段階化を提案している。この手法では、まず大幅な近似で認識を行ってみて、良い結果が得られれば認識処理を終了する。一方、得られなければ、段階的に近似を弱めて認識処理を再度施す。これにより、多段階化を行わない場合と比べて、平均で処理時間が1/10程度に短縮できることが示されている。その結果、データベースに収めた2.6G個の36次元ベクトルと検索質問画像から得られる260個のベクトルが60msで照合できる。

### 5.4.2 省メモリ化

特定物体認識の大規模化で最も深刻な問題はメモリ量に関するものである。まえがきでも述べたように、特徴ベクトルをすべて保存しておく素朴な手法では、メモリの消費が膨大となる。

この問題を解決する一つの方法は、特徴ベクトルそのものを保持しないことである。5.3.3の手法では、ハッシュを引いて最近傍の候補集合を得たのち、距離計算を行ってその中から距離最小のものを選択するが、距離計算を諦めて、すべてに照合するものとして扱えば、特徴ベクトルを保持せずに済む。この場合、特徴ベクトルはハッシュのインデックスとして表されて



おり、インデックスの一致検索によって照合されていると言える。bloomier フィルタという確率的データ構造を用いると、より一層のメモリ節約が実現できることが知られている [31]。

以上の方法は局所特徴量を表す特徴ベクトルの記録を圧縮する方法である。一方、認識対象がさらに大規模化すると、BOF 表現自体の記録が問題となってくる。先に述べた通り、BOF 表現は画像 1 枚に対して 1 つ必要となるため、データベースの規模に対して線形に増加する。

この問題を解決する一つの方法は、BOF 表現をコンパクトにすることである。具体的には、重み情報を捨てて、visual word の存在のみを記録し、ビットベクトルとすること、さらにインデックスの圧縮技術 [13] を導入することなどがある [32]。

それでも不足する場合には、BOF 表現をそのままのベクトルとして持つことを止め、ベクトル量子化して整数値に変換することが試みられている。先に述べた通り、ベクトル量子化によって局所特徴量のベクトルを visual word(の ID) に変換することにより、大幅な圧縮が可能となっている。これと同じことは、BOF 表現のベクトルにも適用できる。局所特徴量の場合と同様、ベクトル量子化を行うと転置インデックスを利用可能となるため、照合も高速化される。この手法は miniBOF と呼ばれている [32]。実際には、整数値だけでは照合の良さが評価できないため、Hamming Embedding と同様の手法によりビットベクトルのシグネチャを生成して併用する。ただし、miniBOF の手法では、Hamming Embedding のように照合の制限にビットベクトルを用いるのではなく、BOF 表現間の距離を推定する目的で用いる。

#### 5.4.3 2 次記憶のための手法

上記の手法の多くは、データを主記憶上に置くことによって、高速な処理を実現している。ところが、利用可能な主記憶の容量をデータ量が上回る場合には、低速な 2 次記憶を用いる必要が生じる。この目的を満たす手法として [33] が提案されている。また、2 次記憶の利用に適した近似最近傍探索の手法として [34] がある。

#### 5.4.4 認識率の向上

DB 画像と検索質問画像の間で照明条件や視点などの撮影条件が大幅に異なる場合、一般に認識率が低下する。このような場合でも認識率を低下させないためには、(1) 撮影条件が変動しても安定的に取り出せる特徴量を開発すること、(2) 検索質問画像を DB 画像に合うように変換すること、(3) あるいはその逆に、DB 画像を検索質問画像に合わせるもののいずれかが必要である。ここでは、(2) と (3) について述べる。

DB 画像を認識対象の標準的な画像とみる場合、検索質問画像を DB 画像に合わせる (2) のプロセスは正規化処理と言える。この処理は認識時に適用する必要があるため、あまり計算コストをかけることができない。また、画像が大幅に劣化している場合には、正規化が困難となる。一方、(3) のプロセスは撮影条件の変動モデルの導入にあたり、予め DB 画像に適用しておくことが可能である。このような観点から視点の変化を扱う手法 [35] や、撮影時のボケ・ブレなどの劣化を扱う手法 [36] が提案されている。これらの手法では、データベースの画像に変動

モデルを適用して変動を受けた画像を数多く生成し、そこから特徴を抽出しておくことにより、変動に対する頑健性を得る。ただし、このプロセスからも分かるように、メモリ量の制限から大規模データベースに対して不用意に導入することはできない。

様々な条件下で撮影された画像がデータベースに格納されており、それらなるべく多く検索したい場合には、上記とは異なるアプローチが可能である。まず第一段階として検索質問画像を用いて対応する DB 画像のうち検索質問画像に近いものを取り出す。そして、得られた DB 画像も加味して、さらに別の DB 画像を取り出す。このような処理は、情報検索における質問拡張 (query expansion) [15] と呼ばれる処理である。画像の検索に対して質問拡張を適用した例が報告されている [37]。

## 6. 検 証

最後に、検証のプロセスについて述べる。これは、照合処理によって得られた候補が正しいかどうかを、別の特徴を吟味することによって確かめる処理である。これまでに述べてきた特定物体認識の手法は、BOF 表現に基づくものであり、局所特徴量が取り出される元となった局所領域相互の幾何的關係は捨てられている。検証処理では、認識対象が剛体であるとの仮定の下、この情報を用いて照合結果が正しいかどうかを確かめる。具体的には、検索質問画像と DB 画像の間で適切な幾何変換が存在するかどうかを確かめる [6]。また、照合の結果、候補に順位が付けられている場合には、検証結果を用いて順位を変更する [17]。

一般に、このような幾何変換の推定は比較的計算時間がかかるため、多数の候補に対して適用することはできない。ここで、目的が幾何変換の推定ではなく、あくまでも順位付けである場合には、もう少し簡易な方法で検証を行うことが考えられる。Weak Geometrical Consistency (WGC) [20] と呼ばれる処理はそのような目的を持つものである。具体的には、局所領域を抽出する際に得られる、局所領域の方向とスケールに着目し、それらが検索質問画像と DB 画像で一貫するかどうかを検証する。このような検証処理は、認識率や順位付けの向上に極めて有効である。

以上の検証処理は、簡易なものであっても照合処理と比べると計算コストがかかる。したがって、検証処理の対象となる候補は絞り込まれたものでなければならない。このような候補のリストは short list と呼ばれることがある。逆にいえば、照合処理と検証処理には次のような役割分担がある。照合処理は計算コストはかからないが、正確性という点では十分とは言えない処理である。ただし、short list の中に十分な数の正解を収めることはできる。一方、検証処理は、その計算コストの高さから DB 画像全体を対象とすることはできないが、十分絞り込まれた short list については適用でき、正解が含まれていればそれを上位に順位付けし直すことが可能である。

メモリ量の点でも、役割分担は明確である。short list を得るまでの照合処理は、DB 画像全体を対象とするためメモリ効率を高める必要があるが、検証処理は、short list 中の限られた候



補のみを対象とするため、その段階で2次記憶から読み出すこととすれば、メモリの制約をあまり受けない処理が可能である。

## 7. む す び

本稿では、物体のインスタンスを認識する技術として局所特徴量を用いた特定物体認識を取り上げ、基本的な技術や最近の研究成果について述べた。特定物体認識の問題を一言でいえば大規模化への挑戦である。局所特徴量を用いた単純な照合と投票による認識でも高い認識率を得ることが可能であるが、認識対象が大規模になるとメモリ量や処理時間の問題が生じて現実的ではなくなる。この問題は、画像表現、索引付け、照合、検証といった処理を工夫することにより、認識率の低下を抑えつつ解決することが可能である。

今後、大規模化がギガからテラオーダに進展すれば、特定物体認識を通して、現在のWebを実世界に拡張することが視野に入ってくる。本稿がそのような発展の一助となれば幸いである。

**謝辞** 本稿の一部は筆者の共同研究者である岩村雅一氏との議論に基づく。また、図の一部も彼による。本稿に掲載した我々の成果の一部は、科学研究費補助金基盤研究(B)(19300062)の補助によるものである。ここに記して感謝する次第である。

## 文 献

- [1] 井手一郎, 柳井啓司, “セマンティックギャップを越えて — 画像・映像の内容理解に向けて —,” 人工知能学会誌, vol.24, no.5, pp.691–699, Sept., 2009.
- [2] 黄瀬浩一, 岩村雅一, “3日で作る高速特定物体認識システム,” 情報処理, vol.49, no.9, pp.1082–1089, Sept., 2008.
- [3] Y. Ke, R. Sukthankar and L. Huston, Efficient near-duplicate detection and sub-image retrieval, Proc. of ACM MM’04, 2004.
- [4] O. Chum, J. Philbin and A. Zisserman, “Near duplicate image detection: min-hash and tf-idf weighting,” BMVC2008, 2008.
- [5] C. Schmid and R. Mohr, “Local grayvalue invariants for image retrieval,” IEEE PAMI, vol.19, no.5, pp.530–535, 1997.
- [6] D. Lowe, “Distinctive image features from scale-invariant keypoints,” International Journal of Computer Vision, vol.60, no.2, pp.91–110, 2004.
- [7] O. Chum, J. Philbin, M. Isard and A. Zisserman, “Scalable near identical image and shot detection,” CVIR’07, pp.549–556, 2007.
- [8] E. Nowak, F. Jurie and B. Triggs, Sampling strategies for bag-of-features image classification, Proc. ECCV2006, Vol. 4, pp.490–503, 2006.
- [9] K. Mikolajczyk and C. Schmid, “A performance evaluation of local descriptors,” IEEE PAMI, vol.27, no.10, pp.1615–1630, 2005.
- [10] 本道貴行, 黄瀬浩一, “大規模画像認識のための局所特徴量の性能比較,” 画像の認識・理解シンポジウム (MIRU2008) 論文集, p.550, July, 2008.
- [11] Y. Ke and R. Sukthankar, Pca-sift: A more distinctive representation for local image descriptors, CVPR2004, Vol. 2, pp.506–513, 2004.
- [12] J. Sivic and A. Zisserman, Video google: A text retrieval approach to object matching in videos, Proc. ICCV2003, Vol. 2, pp.1470–1477, 2003.
- [13] C. D. Manning, P. Raghavan and H. Schütze, Introduction to Information Retrieval, Cambridge, 2008.
- [14] I. H. Witten, A. Moffat and T. C. Bell, Managing Gigabytes — Compressing and Indexing Documents and Images, Second Edition, Morgan Kaufmann Publishers, 1999.
- [15] R. Baeza-Yates and B. Ribeiro-Neto, Modern Information Retrieval, Addison Wesley, 1999.
- [16] D. Nistér and H. Stewénius, Scalable recognition with a vocabulary tree, Proc. CVPR2006, pp.775–781, 2006.
- [17] J. Philbin, O. Chum, M. Isard, J. Sivic and A. Zisserman, Object retrieval with large vocabularies and fast spatial matching, Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp.1–8, June, 2007.
- [18] 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, Aug., 2009.
- [19] H. Jégou, M. Douze and C. Schmid, Searching with quantization: Approximate nearest neighbor search using short codes and distance estimators, INRIA Technical Report 7020, 2009.
- [20] H. Jégou, M. Douze and C. Schmid, Hamming embedding and weak geometric consistency for large scale image search, Proc. ECCV2008, 2008.
- [21] J. Philbin, O. Chum, M. Isard, J. Sivic and A. Zisserman, Lost in quantization: Improving particular object retrieval in large scale image database, Proc. of CVPR2008, 2008.
- [22] G. Shakhnarovich, T. Darrell and P. Indyk Eds., Nearest-neighbor methods in learning and vision, The MIT Press, 2005.
- [23] 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.
- [24] A. Andoni, M. Datar, N. Immorlica, P. Indyk and V. Mirrokni, Locality-sensitive hashing using stable distributions, Nearest-Neighbor Methods in Learning and Vision (Eds. by G. Shakhnarovich, T. Darrell and P. Indyk), The MIT Press, pp.61–72, 2005.
- [25] A. Andoni and P. Indyk, “Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions,” Comm. of the ACM, vol.51, no.1, pp.117–122, 2008.
- [26] 松下裕輔, 和田俊和, “一般分布に対する principal component hashing,” 情報処理学会研究報告, pp.283–288, Mar., 2008.
- [27] R. Salakhutdinov and G. Hinton, “Semantic hashing,” Proc. SIGIR Workshop on Information Retrieval and Applications of Graphical Models, 2007.
- [28] Y. Weiss, A. Torralba and R. Fergus, “Spectral hashing,” Advances in Neural Information Processing Systems, 2008.
- [29] D. M. Mount, Ann programming manual, 2005. <http://www.cs.umd.edu/~mount/ANN/>.
- [30] 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高速特定物体認識,” 電子情報通信学会論文誌 D, vol.J92-D, no.12, Dec., 2009.
- [31] K. Inoue and K. Kise, “Compressed representation of feature vectors using a bloomier filter and its application to specific object recognition,” Proceedings of the 1st International Workshop on Emergent Issues in Large Amount of Visual Data (WS-LAVD2009), pp.2133–2140, Oct., 2009.
- [32] H. Jégou, M. Douze and C. Schmid, Packing bag-of-features, Proc. ICCV2009, 2009.
- [33] F. Fraundorfer, H. Stewénius and D. Nistér, A binning scheme for fast hard drive based image search, Proc. CVPR2007, pp.1–6, 2007.
- [34] N. Himeí and T. Wada, Approximate nearest neighbor search on hdd, Proc. WS-LAVD, 2009.
- [35] V. Lepetit and P. Fua, “Keypoint recognition using randomized trees,” IEEE PAMI, vol.28, no.9, pp.1465–1479, 2006.
- [36] 黄瀬浩一, 野口和人, 岩村雅一, “参照特徴ベクトルの増大による特定物体認識の高速化と高精度化,” 画像の認識・理解シンポジウム (MIRU2009) 論文集, pp.335–342, July, 2009.
- [37] O. Chum, J. Philbin, J. Sivic, M. Isard and A. Zisserman, Total recall: Automatic query expansion with a generative feature model for object retrieval, Proc. ICCV2007, 2007.