3日で作る高速特定物体認識システム*

黄瀬 浩一　　岩村 雅一
（大阪府立大学）

はじめに

物体認識（object recognition）とは，画像に写っているものが何であるのかを言い当てる処理である．画像を認識することに焦点を当てる場合には，画像認識とも呼ばれる．処理の目的が単純明快であり，誰にでも容易に理解できる反面，人間と同じレベルの認識能力を実現することはすこぶる難しい．このような「単純だが奥が深い」という問題は研究者を魅了しやすいが，実際その通りであり，これまでに数多くの研究が行われてきた．

最近，物体認識は新たなブームとでも呼ぶべき状況にある．たとえば，本年6月上旬に開催されたSSII2008の特別セッション「一般物体認識にチャレンジ」[1]では，開催最終日の最終セッションであったにもかかわらず，多数の参加者があった．また，同月下旬に行われたIEEE CVPR（Computer Vision and Pattern Recognition）[2]では，物体認識に関連する多数の発表があった．皆が夢を忘れていない証拠である．

ところで，一口に物体認識といっても，すべてが手に負えないほど難しいというわけではない．物体認識に必要な技術の飛躍的な進歩によって，ある特定の物体が画像に写っているかどうかを言い当てる処理（特定物体認識あるいは物体検出と呼ばれる処理）は，コンピュータによる実現やその商用利用が可能なものとなってきた．

筆者が所属する大阪府立大学工学部知能情報工学科では，3年生を対象としてミニ卒業研究形式の学生実験を行っている．そのテーマの1つが特定物体認識システムを作成するというものである．物体認識システムを作るというと，いかにも大変な実験のように思われるかもしれないが，実際には，公開されているいくつかのソフトウェアを連結し，それにインタフェースをかぶせるという単純なものである．前提条件としては，学生はC言語とC++言語が理解でき，ある程度プログラムも書けること，Windows PC（メモリ1GB以上）とWebカメラが利用可能であること，開発環境としてVisual Studio 2005あるいは2008が利用可能であることを想定している．

プログラミングにそれほど自信がない学生でも，物体認識の原理を理解しつつ成果報告を完了できるようにするため，実験の期間を半年としている．しかし，プログラミングの作業量に限って言えば，ある程度の心得のある人ならば3日程度のものである．本稿では，学生実験の経験を踏まえつつ，3日間でどのように特定物体認識システムが作成できるのかについて紹介したい．

一般物体認識 vs. 特定物体認識

先に述べたように，物体認識は研究者を魅了するが，その面白さ，困難さを一般の人に伝えることは容易ではない．理由は，物体認識という処理が我々人間にとってあまりにも自然なタスクであることと考えられる．幼児であっても，椅子，机，自動車，動物など，実に多様な物体をいとも簡単に認識することができる．認識するにあたって，何かを意識的に考えたり，迷ったりすることはほとんどない．そのため，専門知識のない一般の人には，なぜコンピュータによる物体認識がそれほど難しいのか，容易には理解できない．このような，コンピュータが不得意な物体認識は，椅子，自動車など，一般的な物体のカテゴリを言い当てる処理であり，一般物体認識（generic object recognition）と呼ばれている．

一方，一般の人が異口同音に驚愕するのは，人間が容易にできないことをコンピュータが簡単に行う場合であ

る．最近の事例では，インターネットの検索エンジンの処理を挙げることができる．世界人口をはるかに上回る数の Web ページから，ユーザが望むものを瞬時に探し出す処理は，人間が到底まねのできないものである．画像に関する場合でも同様であり，たとえば，非常によく似た画像を，膨大な画像のデータベースから発見する処理（near duplicate detection）は，人間には容易ではないがコンピュータには比較的容易な処理である．物体認識の場合であっても，手元の画像に写っている物体とまったく同一のものが写った画像を，膨大な画像のデータベースの中から探し出す処理は，コンピュータにとって比較的容易である．このように，コンピュータが得意な物体認識は，まったく同じ物体かどうかを言い当てる処理であり，特定物体認識（specific object recognition）と言われている．

一般物体認識との本質的な差異は，認識対象の形状に生じ得る変動のレベルにある．たとえば，同じ椅子と呼ばれる物体でも，実に多様な形状のものが存在し得るが，特定物体の場合は同じものであるため，対象物体が剛体ならば形状の変動は存在しない．人間は多様な変動を柔軟に扱う処理が得意であるが，コンピュータは変動の少ない対象であれば，大規模な選択肢の中から高速に認識することが得意であると言える．

本稿では，特定物体を対象として認識システムを構築することを考える．これは，専門家というよりは一般の人に近い感覚を持つ学生に，物体認識への興味を持たせる入り口としては，こちらの方が適していると考えるからである．なお，以後は混乱のない場合，特定物体認識を単に物体認識と呼ぶこともある．

認識システムの狙いと構成

3 日間で認識システムを作成する方法について述べる前に，システムの狙いと構成についてまとめておく．

ここで対象とする特定物体認識は，学生実験としても無理なく実施できるように単純なものとする．具体的には，ポスターや写真などの平面物体を認識対象とし，入力画像には，複数の平面物体が含まれていることはないと仮定する．ただし，画像には平面物体全体が含まれているとは限らず，一部の可能性もある．さらには，撮影する角度や照明の条件などには特に強い制限を設けず，手持ち撮影するような状況を想定する．

図 -1 に，特定物体認識システムの構成を示す．特定物体認識システムは 3 つのソフトウェアモジュールと 1 つのデータベース（物体モデル）を組み合わせることで構成される．

Web カメラを通して撮影された未知の画像（以後，クエリ画像と呼ぶ）は，ユーザインタフェースを経て特徴抽出モジュールに送られる．特徴抽出モジュールでは，入力画像から認識に用いる特徴を抽出する．特徴としては，画像の特徴的な一部分から抽出される局所特徴量と呼ばれるものを用いる．認識対象となる平面物体からはあらかじめ特徴を抽出しておき，物体モデルとして登録しておく．特徴照合モジュールでは，クエリ画像から抽出された局所特徴量と，物体モデルに収められた局所特徴量を照合し，認識結果を得る．それを受けて，ユーザインタフェースでは認識結果をユーザに表示する．

本来は，これらのモジュールを作成し，連結することにより，認識システムを作成することができる．しかしながら，作成する際の面白さを考えて，ここでは，以下の 3 段階でシステムを作成していく．本稿では，この各段階が 1 日分の作業量であると見積もっている．

(1) 低速特定物体認識システムの構築(1 日目)

まずは特徴抽出モジュールと物体モデルの作成に主眼を置き，動作確認のために簡単な特徴照合モジュールを作成して組み合わせる．これにより，低速ではあるものの認識システムを構築することができる．

(2) 近似最近傍探索を用いた高速化(2 日目)

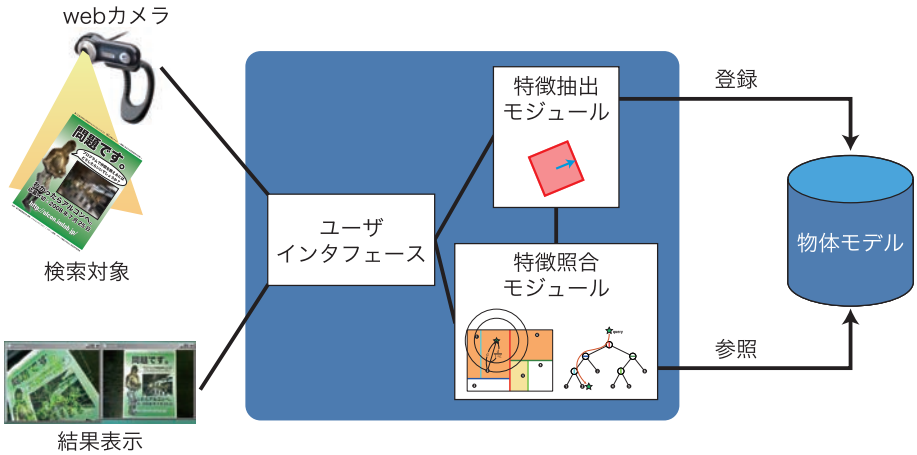

図 -1　物体認識システムの構成

上記システムを高速化するために，近似最近傍探索と呼ばれる手法を導入した特徴照合モジュールを構築し，置き換える．これにより高速化を図る．

(3) ユーザインタフェースの構築(3 日目)

Web カメラからの画像入力や結果出力のインタフェースを整えることにより，実時間で対象を認識するシステムとして完成させる．

以下，順に述べていく．

低速特定物体認識システムの構築

局所特徴量

一般物体認識と比べて特定物体認識が容易な理由は，対処すべき物体形状の変動が少ないことである．特に平面物体を対象とした場合には，対象自体の形状の変動は存在しない．それでも，十数年前は，特定物体認識ですら実現することが困難であった．その理由は，形状の変動以外にも画像にはさまざまな変動があり，それに有効に対処する術がなかったことである．

代表的な変動要因を**図 -2** に示す．図-2(a)の画像が(b)～(d)の変動を受けた例を示している．(b)の幾何学的変動とは，対象とカメラの位置関係が定まらないことによって生じる問題である．(c)の照明変動は，認識対象の周囲にある光源や他の物体による影が変動することによって生じる．(d)のオクルージョンは，他の物体によって認識対象の一部分が隠れることを意味する．認識対象の全体を撮影せず，一部分だけを撮影する場合にも同様の問題が生じる．

(a) 元画像　(b) 幾何学的変動

(c) 照明変動　(d) オクルージョン

図 -2　画像の変動要因

これらの変動要因によって，同じ物体が画像中に写っていたとしても，画像としては大きく異なるものになり得る．認識に用いる画像の特徴量は，画像から抽出するものであるため，変動の影響を受けて値が変化することが十分考えられる．たとえば，画像全体から特徴量を抽出するという方法を用いる場合，オクルージョンが生じると対象物体の隠れた部分からは特徴量を抽出できなくなるため，結果として得られる特徴量は異なったものとなってしまう．

ところが近年，この問題を解決する新しい特徴抽出法が提案された．これは，画像を解析することによって特徴的で局所的な領域を求め，その領域を変動に対してロバストな方法で記述するものである．どのような角度から撮影しても，常に同じ局所領域が得られるならば，幾何学的変動の影響は受けない．また，局所領域の輝度を正規化することにより，照明変動の影響も限定的なものとなる．さらに，オクルージョンの影響を受けていない(隠れていない)部分から局所領域が抽出される限り，対象を認識できる可能性がある．実際には，各種変動にまったく影響を受けないということはないが，影響を限定的にすることが可能になっている．

SIFT の利用

局所特徴量の中でも，最も有効な手法の 1 つとして知られているのは，Lowe による SIFT (Scale-Invariant Feature Transform)[3] である．そこで，本システムでは，これを利用する．

図 -3 に SIFT で得られる局所領域の模式例を示す．図中の正方形が局所領域を表し，矢印が局所領域の向きを表す．混乱を避けるため，この図では局所領域を 1 つしか描いていないが，実際には数百から数千の局所領域が抽出される．図 -3 の (a) と (b) を比べると分かるように，撮影対象の大きさや向きが変わっても，物体からは同じ局所領域が取られている．局所領域が正方形であるため，SIFT で対応可能な幾何学的変換は，相似変換(拡大，縮小，並進，回転)であり，図 -2 (b) のように図の形が変わるようなもの(射影変換など)には対処できない．ただし，ほぼ正面から撮影する場合には，相似変換の範囲で近似できることも多く，実際に有効である．個々の局所領域からは，照明の変動に対して安定した 128 次元の特徴ベクトルが局所特徴量として抽出される．

局所領域の定め方や，そこからの特徴抽出の方法については，優れた解説がすでにいくつか公表されている．その中でも，藤吉によるもの[4]は分かりやすい．英

語になるが短い解説やさまざまな資料へのポインタは，Wikipedia にもある[5]．こちらも適宜参照されたい．

本システムでは，上記の Wikipedia からもリンクが張られている Hess によるソースコード[6]を利用する．利用のための準備として，まず OpenCV[7), 8)]と呼ばれるライブラリをインストールする．次に，Hess によるソースコードを取得し，siftFeat をコンパイルしてみる．OpenCV の設定などが正しくなされていればコンパイルは成功し☆1，実行すると特徴抽出の結果を表示できるはずである．特徴抽出のための関数 _sift_features は，sift.c に定義されている．使い方は，siftfeat.c 中の main に記されている通りであり，入力画像 img から抽出された特徴ベクトルが features という変数に格納される．

物体モデルの構築

認識システムを構築する上で，次に行うことは，認識対象となる平面物体のモデルを作成することである．SIFT のプログラムは，与えられた画像から特徴ベクトルを抽出するだけであるため，どの特徴ベクトルがどの物体から得られたものであるかを，別途記録しておく必要がある．ここでは，平面物体の画像から抽出された特徴ベクトルを物体 ID とともに記録するという方法を採用する．具体的には，SIFT で抽出される 128 次元の特徴ベクトルを＜特徴ベクトル＞と表すとき☆2，各特徴ベクトルがどの物体から抽出されたのかを表すために＜物体 ID＞を追記しておく．つまり，

＜物体 ID＞＜特徴ベクトル＞

という記述を特徴ベクトルの数だけ用意して，ファイルに保存する．この処理を行うプログラムは，前述の siftfeat.c を参考にすれば簡単に作成できる．

画像の複雑さにもよるが，VGA 程度の解像度であれば，画像 1 枚から通常は数百から数千の特徴ベクトルが抽出される．たとえば，特徴ベクトルの数が画像あたり 2 千個とすると，平面物体 1 つあたり（画像 1 枚あたり）の記憶容量が約 500KB となる．この値から，物体モデルとして記録することのできる平面物体数の目安が分かる．

☆1 siftFeat のコンパイルには OpenCV が必要となる．コンパイルができない場合，Visual Studio の設定やプロジェクトの設定が文献 8) に記載の通りにされているかどうかをチェックするとよい．

☆2 SIFT によって抽出されるデータは，128 次元の特徴ベクトルに加えて，特徴ベクトルの数や局所領域の座標，方向などがある．ここでは，これらのデータは不要なので扱わない．

(a) (b)

図 -3 SIFT で得られる局所領域

最近傍探索による物体認識

物体モデルが用意できれば，クエリ画像から得た特徴ベクトルと物体モデルの特徴ベクトルを照合することによって，物体認識システムを作ることができる．照合方法として，ここでは網羅的な最近傍探索を用いる．

いま，未知の画像から抽出した特徴ベクトルを $\boldsymbol{q}$，物体モデルに記録された特徴ベクトルを $\boldsymbol{p}_i$ とし，$\boldsymbol{q}$ と $\boldsymbol{p}_i$ のユークリッド距離を $d(\boldsymbol{q}, \boldsymbol{p}_i)$ とする．ここでは，すべての i について距離を計測し，最近傍となる $\boldsymbol{p}_{i^*}$（距離が最小のもの）を求める．そして，$\boldsymbol{p}_{i^*}$ の物体 ID を調べ，その物体に 1 票を投じる．クエリ画像から得たすべての特徴ベクトルについて以上の処理を行い，得票数最大となった物体を認識結果とする．

実際にプログラムを走らせてみると，このような単純な処理であっても驚くほど正確に物体を認識できることが分かるが，改善すべき点も多い．特に処理時間の問題は大きい．物体モデルに記録された特徴ベクトル $\boldsymbol{p}_i$ の数が膨大になるため，距離計算のような単純な処理であっても，長い時間がかかってしまう．

たとえば，未知画像から得た特徴ベクトルの数を 600，物体モデルに含まれる物体数を 1 万，特徴ベクトルの合計を 2 千万（物体あたり 2000）として距離計算を行うと，最近のコンピュータを用いても 1 枚の画像を認識するのに数十分以上の時間が必要となる．すなわち，単純な最近傍探索では低速な特定物体認識しか実現できないと言える．

近似最近傍探索を用いた高速化

最近傍探索の高速化

物体認識を高速化するにはどうすればよいであろうか．最も処理時間を要する部分は最近傍探索であるので，これの高速化を考えることは順当であろう．

最近傍探索の高速化には，大きく分けて 2 通りの方法がある．1 つは個々の距離計算自体を高速化する方法，もう 1 つは，距離計算の回数を減らす方法である．

距離計算自体を高速化する方法には，距離計算を途中

で打ち切るものがある．いま，物体モデルに記録されたいくつかの特徴ベクトルと距離計算が終了し，それまでの最小値が $d_{\min}$ であるとしよう．そして，$\boldsymbol{q}=(q_1,\ldots,q_n)$ と次の対象 $\boldsymbol{p}_i=(p_{i1},\ldots,p_{in})$ の距離計算を行う途中で，

$$\sum_{j=1}^{m}(p_{ij}-q_j)^2 > d^2_{\min} \quad (m<n)$$

が満たされたとする．この場合，$\boldsymbol{p}_i$ はもはや $\boldsymbol{q}$ の最近傍になることはないため，計算を打ち切って，次の特徴ベクトルとの距離計算に移ることができる．

この方法は確かに高速化に寄与するが，ベクトルの次元数 n に対して，$1/n$ より処理時間を短縮することはできないので，次元数がそれほど大きくない場合には，効果は限定的と言える．

もう 1 つの方法，すなわち距離計算の回数を減らす方法には，距離計算の対象となる特徴ベクトルを絞り込むものがある．距離計算の対象が少なければそれだけ，高速化が望める．絞り込みの戦略には，必ず正しい最近傍が得られる手法と，そうとは限らない手法がある．

前者は安心して適用できる反面，効果が限定的であり，大幅な高速化は望めない．また，ベクトルの次元が増えれば増えるほど効果が低くなり，20 ～ 30 次元を超えるあたりで効果がなくなる（すべてのベクトルと距離計算をする手法と同程度以上の時間がかかる）場合が多い．

一方，後者は近似最近傍探索と呼ばれる手法であり，正確性を犠牲にすることによって，数千倍から数万倍といった劇的な高速化を図るものである．「最近傍探索による物体認識」のところで述べたように，我々の物体認識システムでは，認識結果は投票によって決まる．このような場合，個々の投票がある程度，不正確であったとしても，全体として正しい投票が多ければ物体を正しく認識できる．そこでここでは，近似最近傍探索の手法を導入し，高速化を図ることにしよう．

近似最近傍探索

近似最近傍探索の代表的な手法には，木構造を用いるものとハッシュを用いるものがある．木構造を用いる手法の代表は ANN（Approximate Nearest Neighbor）[9),10)]，ハッシュを用いる手法の代表は LSH（Locality Sensitive Hashing）[11),12)] と言える．本稿では，このうち，理解がより簡単な ANN を取り上げる．

ANN は，単一のデータ構造やアルゴリズムを表すものではなく，複数のデータ構造とそれを扱うためのさまざまなアルゴリズムのライブラリである．ここでは，データ構造として kd-tree（k-dimensional tree）を使った最も単純な方法を用いることにする．

図 -4 に kd-tree の概念図を示す．図 -4 (a) が特徴空間，(b) がそれに対応する kd-tree である．kd-tree では，次のような再帰的な処理により特徴空間を分割して木構造を構成する．最初は，すべての特徴ベクトルを囲む超長方形（hyperrectangle）を考え，それに対応する根ノードを 1 つ設ける．次に，特徴空間の 1 つの次元に閾値を設け，空間を 2 分割する．分割した各々の超長方形について，同様の処理を繰り返し，超長方形に含まれる特徴ベクトルが 1 つになった段階で停止する☆3．最終的に得られる領域（葉ノードに対応するもの）を以下ではセルと呼ぶ．

このようにして構成された木構造を用いて，近似ではない最近傍探索を行う方法から説明しよう．

☆3 分割する次元と閾値を決める方法にはさまざまなものがある．詳しくは文献 9) を参照のこと．

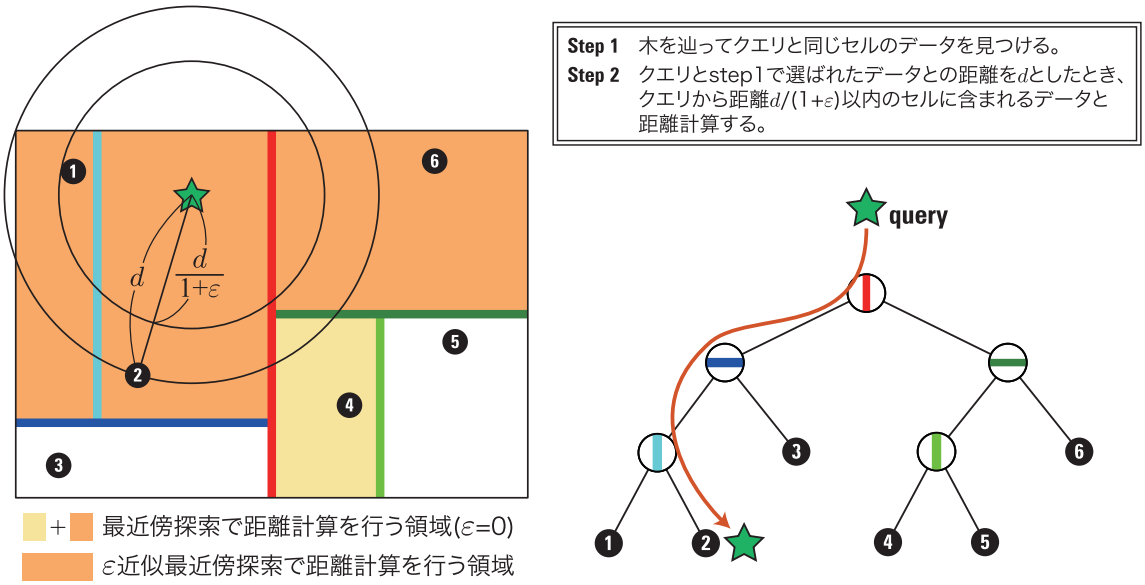

図 -4 kd-tree

まず Step1 として，クエリ画像の特徴ベクトル $\boldsymbol{q}$ が，どのセルの内部にあるのかを，木の探索によって求める．具体的には，根ノードから順に，ノードが表す次元を見て，閾値との大小関係により，どちらを辿るかを決めて，セル(葉ノード)まで降りればよい．

発見したセルに対応付けられている特徴ベクトルを $\boldsymbol{p}$ とするとき，真の最近傍は，$d(\boldsymbol{p}, \boldsymbol{q})$ を半径とする超球(hypersphere) の中にある．そこで Step2 では，超球と重なりを持つ他のセルを訪問し，そのセルに含まれる特徴ベクトルとの距離を計算する．$\boldsymbol{q}$ の最近傍は，最小距離を与える $\boldsymbol{p}$ となる．

以上の処理に近似最近傍探索を導入する方法は，非常に直接的である．セルの訪問を制御する距離 d をそのまま用いるのではなく，$1/(1+\varepsilon)$ を乗じ，半径を小さくして処理を行えばよい．ここで $\varepsilon \geq 0$ である．これによって距離計算の対象となる特徴ベクトルを削減することができるために高速化が可能となる．ただし，削減したものの中に真の最近傍が含まれていた場合，正しい最近傍は求められなくなる．

近似最近傍探索の導入

ANN のライブラリは文献 10) からダウンロードできる．ANN の使い方については，提供されているサンプルコード “ann_sample.cpp” が参考になる．以下，物体認識に用いるために改変すべきポイントについて述べる．

ANN は kd-tree に記録された特徴ベクトルのうち，クエリの特徴ベクトルの近似最近傍となるものを求めるプログラムである．したがって，特徴ベクトルと物体 ID の対応関係を用いて近似最近傍の特徴ベクトルから物体 ID を求める部分，物体に対して投票処理を行う部分，さらに得票数から認識結果を決定する部分については，独自に作成しなければならない．

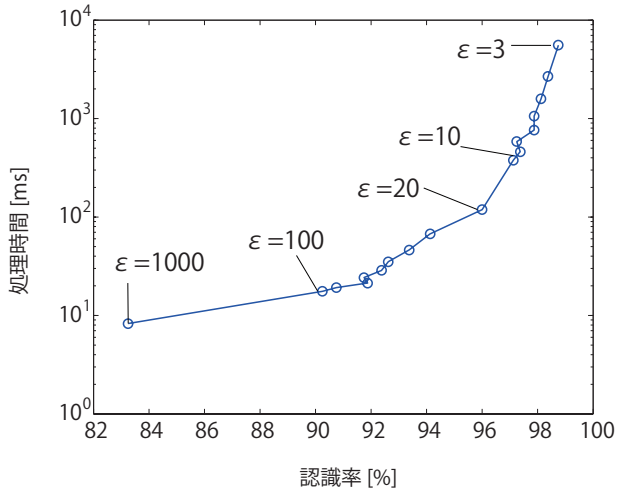

図 -5　認識率と処理時間の関係

まず，物体モデルの特徴ベクトルを kd-tree に格納するところから始めよう．「物体モデルの構築」のところで述べたように，各特徴ベクトルには，物体 ID が付けられている．特徴ベクトルが記録されている順番をその特徴ベクトルの ID とすると，まず，物体モデルを読み込む際に，特徴ベクトル ID と物体 ID の対応関係を対応表に記録する．次に，128 次元の特徴ベクトルを kd-tree に記録する．これには ANNkd_tree というコンストラクタを用いる☆4．

この処理が終われば，近似最近傍探索が可能な状態となっている．クエリ画像から抽出した個々の特徴ベクトルに対して，先の網羅的な探索のかわりに，annkSearch というメソッドにより近似最近傍となる特徴ベクトルの ID を求める☆5．そして，物体 ID との対応表を参照することにより，対応する物体に投票する．すべての投票が終了すると，得票数最大のものを結果として出力する．

処理時間は ε (プログラム中では eps という変数に対応) の値によって大きく変化する．ε の値と処理時間，認識率の関係を**図 -5** に示す．このグラフは，物体モデル数を 1 万，クエリ画像数を 800 として計測したものである．ε は 3 ～ 1000 まで変化させている．この物体モデルとクエリ画像について言えば，認識率をあまり落とさずに処理時間を改善する値としては，ε=20 程度がよいと思われる．このとき特徴ベクトルの照合には，クエリ画像 1 枚あたり 120ms を要し，そのときの認識率は 96.0% である．先に述べた網羅的な探索の場合と比較すると，数万倍の高速化を達成している．

ユーザインタフェースの構築

画像サイズが QVGA 程度であるならば，SIFT による局所特徴量の抽出に必要な時間が数百ミリ秒にとどまるため，照合と合わせても 1/2 秒程度の処理時間となる．これによって，撮影した画像をその場で認識して結果を表示することが可能となる．ここでは，そのための処理の方式とユーザインタフェースの構築について紹介しよう．

処理の方式として最も単純なのは，過去の認識結果を一切顧みず，現在，画像に捉えている対象を常に認識し，結果を返すものである．ここでは単純化のため，この方式を採用する．

次に，ユーザインタフェースについて述べる．これま

☆4 引数のうち，dataPts には特徴ベクトルのリストを入れておく．nPts は特徴ベクトルの総数，dim は 128 である．

☆5 引数のうち，queryPt にはクエリとなる特徴ベクトルを格納する．k は求める最近傍の数であり，ここでは 1 である．nnIdx は特徴ベクトルの ID を返すための変数である．eps については後述する．

で述べてきた処理とは異なり，ユーザインタフェースの構築はOSやハードウェアに依存したものとなる．ここでは，安価なWebカメラをWindows PCと組み合わせる場合に焦点を絞り，紹介する．

図-6に画面の構成を示す．Webカメラで撮影された入力画像を表示するウィンドウ(左側)と，認識結果を表示するウィンドウ(右側)の2つを設けている．入力画像のウィンドウには，SIFTで得られた局所領域を，矩形と矢印によって表示している．

このようなインタフェースは，DirectXとDirectShowを組み合わせることによって実現できる．サンプルプログラムは文献13)からダウンロードできる．このプログラムは，Webカメラを通して取得した画像を表示するものであり，Webカメラの解像度とその設定番号のリストを調べる機能，画像取得を一時停止する機能や，取得した画像をファイルに保存する機能などが実現されている．

このプログラムに，物体モデルの読み込み，特徴抽出モジュール，特徴照合モジュールを加えて，認識結果を表示するように変更すれば，認識システムが完成する．物体モデルの読み込みは，プログラムの最初の部分で行っておく．特徴抽出モジュールや特徴照合モジュールは，先に取り上げたものであり，画像取得のためのfor文の中で用いる．認識結果の物体IDを得たあとは，図-6右に示すように，ユーザインタフェースにおいて物体IDに対応する画像を表示する．たとえば，物体IDと画像ファイル名との対応表をあらかじめ用意しておけば，認識結果の物体IDを画像ファイル名に変換してファイルを読み込み，表示することができる．

図-6左に示すように，入力画像に局所領域を表示するためには，Hessによるソースコード[6]のsiftFeatに含まれている，draw_featuresという関数を改変して組み込めばよい．

学生実験としての実施

準備

最後に，学生実験として実施するためのヒントをまとめておこう．

まず必要な機材であるが，通常のPCとWebカメラがあれば十分である．PCに必要なメモリは，認識対象の物体数によって異なるが，物体数を1000とする場合には，おおよそ1GBのメインメモリが必要となる．

開発環境としては，ユーザインタフェースを含める場合には，WindowsをOSとするPCと，Visual Studio .NET（2005か2008）が必要となる．ちなみに，プログラムの動作チェックは，Windows XPならびにVistaのVisual Studio 2005で行った．

もう1つ，準備を要する重要な項目として，画像データの収集がある．高速化の劇的な効果を見る上では，物体モデルを作成するための画像の枚数が数百枚以上あることが望ましい．

スケジュール

次に，実験のスケジュールについて述べる．半年間をかけてミニ卒業研究形式で行う場合の一例を**表-1**に示す．初回は物体認識全般についての導入と関連技術の紹介である．学生の興味を引くために，関連技術としてたとえば，パノラマ画像の生成[14]，フォトツーリズムなどの技術[15)〜17)]，一般物体認識などを紹介するのもよい．

第2〜5週は近似最近傍探索を用いない低速特定物体認識システムの構築である．4週間で原理を理解しつ

第1週	イントロダクション
第2〜5週	低速特定物体認識システムの構築
第6〜9週	近似最近傍探索を用いた高速化
第10〜12週	ユーザインタフェースの構築
第13〜14週	プレゼンテーション準備
第15週	プレゼンテーション

表-1　実験のスケジュール

図-6　インタフェース画面の構成

つプログラムを作成する．手軽にSIFTの感触を掴むのであれば，文献18)で提供されているプログラムを試してみてもよい．

第6～9週は近似最近傍探索を用いた高速化である．こちらについては，残念ながら日本語による初心者向けの解説はなさそうである．現在のところ，ANNのマニュアル[9]が一番分かりやすい．関連する資料としては，和田による解説[19]のほか，本[20]が出版されている．

第10～12週はユーザインタフェースの構築である．ライブラリの概要を理解した上で動くものを作ることに主眼を置けば，3週間で十分構築できる．

最後の3週は，実験結果のプレゼンテーションにあてる．実験を通して得たことをプレゼンテーションと配布資料にまとめて，最終週に発表する．

検討課題

本稿で述べた認識システムは，短期間で無理なく作成するために，単純化した個所が多数ある．学生実験の検討課題としては，これらの個所を取り上げて性能を向上させることが考えられる．

性能の向上には，認識率の向上，処理時間の短縮，メモリ使用量の削減の3点がある．たとえば，SIFTのロバスト性をさらに向上させた手法として，PCA-SIFT[21]が知られている．PCA-SIFTを用いると，特徴ベクトルの次元数を削減できるため，認識率の向上とともに，メモリ使用量の削減も可能となる．処理時間の短縮については，「最近傍探索の高速化」において紹介した距離計算の打ち切りや，一定数の票が集まったときに投票処理を打ち切ることにより，比較的簡単に達成できる．また，物体モデルに記録する特徴ベクトルを削減することも考えるとよい．ただし，これらの手法の中には，導入と引き替えに性能の低下を招くものもある．また，近似最近傍探索のもう1つの代表的な手法としてLSHがある．これを適用して性能や動作の違いについて検討することも，面白いのではないかと思われる．

さらなる高速化や高精度化を目指すのであれば，野口らによる手法[22]が参考になるかもしれない．

おわりに

本稿では，大阪府立大学で行っている学部学生実験をもとに，3日程度で特定物体認識システムを構築する方法について述べた．OpenCVなど数多くのライブラリが利用可能である現在，従来は高度であると思われていた物体認識という処理が，限定的ではあるにせよ，簡単に試してみることができるようになってきた．実際に作ってみるという経験を通して，画像の認識・理解の分野に興味を持つ学生や技術者の皆さんが1人でも増えることを願っている．

謝辞　本実験の実施やサンプルプログラムの作成に協力してくれた，大阪府立大学大学院工学研究科客員研究員（日本学術振興会特別研究員）の中居友弘博士，ならびに同博士前期課程2年の野口和人氏に感謝する．

参考文献

1) 一般物体認識にチャレンジ，第14回画像センシングシンポジウムダイジェスト集（2008）．http://www.vision.cs.chubu.ac.jp/ssii08/
2) Proc. of IEEE International Conf. on Computer Vision and Pattern Recognition (2008).
3) Lowe, D.：Distinctive Image Features from Scale-Invariant Keypoints, International Journal of Computer Vision, Vol.60, No.2, pp.91-110 (2004).
4) 藤吉弘亘：Gradientベースの特徴抽出―SIFTとHOG―，情報処理学会研究報告 CVIM 160, pp.211-224 (2007). http://www.vision.cs.chubu.ac.jp/SIFT/
5) Scale-Invariant Feature Transform, Wikipedia, http://en.wikipedia.org/wiki/Scale-invariant_feature_transform/
6) Hess, R.：A C Implementation of a SIFT Image Feature Detector, http://web.engr.oregonstate.edu/~hess/index.html
7) OpenCVプログラミングブック，毎日コミュニケーションズ（2007）．
8) http://chihara.naist.jp/people/2004/kenta-t/OpenCV/pukiwiki/
9) Mount, D.：ANN Programming Manual, http://www.cs.umd.edu/~mount/ANN/
10) http://www.cs.umd.edu/~mount/ANN/
11) Andoni, A. and Indyk, P.：Near-optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions, Comm. of the ACM, Vol.51, No.1, pp.117-122 (2008).
12) http://www.mit.edu/~andoni/LSH/
13) http://imlab.jp/IPSJ_3days/
14) http://user.cs.tu-berlin.de/~nowozin/autopano-sift/
15) http://phototour.cs.washington.edu/
16) http://labs.live.com/photosynth/
17) http://www.panoramio.com/
18) http://www.cs.ubc.ca/~lowe/keypoints/
19) 和田俊和：空間分割を用いた識別と非線形写像の学習：(1) 空間分割による最近傍識別の高速化，情報処理，Vol.46, No.8, pp.912-918 (Aug. 2005).
20) Shakhnarovich, G., Darrell, T. and Indyk, P. (eds.)：Nearest-Neighbor Methods in Learning and Vision, MIT Press (2005).
21) Ke, Y. and Sukthankar, R.：PCA-SIFT：A More Distinctive Representation for Local Image Descriptors, Proc. CVPR2004, Vol.2, pp.506-513 (2004).
22) 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).

（平成20年7月14日受付）

黄瀬 浩一（正会員） kise@cs.osakafu-u.ac.jp

1986年大阪大学工学部通信工学科卒業．1988年同大学院博士前期課程修了．1990年大阪府立大学工学部電気工学科助手．現在，同大学院工学研究科教授．博士（工学）．2000～01年ドイツ人工知能研究センター客員教授．2006年電子情報通信学会論文賞，2007年IAPR/ICDAR Best Paper Award各受賞．文書画像解析，画像認識，情報検索などの研究に従事．

岩村 雅一（正会員） masa@cs.osakafu-u.ac.jp

1998年東北大学工学部通信工学科卒業．2003年同大学院博士課程後期3年の課程修了．同年，同助手．2004年大阪府立大学大学院工学研究科助手（現在，助教）．博士（工学）．2006年電子情報通信学会論文賞，2007年IAPR/ICDAR Best Paper Award各受賞．パターン認識，コンピュータビジョン，情報検索に関する研究に従事．