データベースシステム 116－2
（1998. 7. 8）

画像類似検索における
複数特徴量間の重みを考慮したアクセス方法

谷口 展郎　　山室 雅司

NTT情報通信研究所

画像や音声といったインフォメーションリッチなメディアのデータに対する検索手段として，原データからその特徴を数値化された「特徴量」として抽出し，与えられた検索キーとの特徴量の類似性に基づいて検索を行う，「特徴量類似検索」が注目されている。我々は，画像検索システム ExSight + HyperMatch における，複数の特徴量を重みづけして検索できる類似画像検索手法についての研究を通じて，最終的な検索結果の精度をほとんど低下させずに，検索時に与えられる重みをもとに，各特徴量における検索精度を犠牲にして検索速度を向上させる2つの手法を考案した。またこのうち1つについては実験を通じてフィージビリティを確認した。

High Speed Access Methods
for Similar Image Retrieval with Feature Weighting

Noburou Taniguchi　　Masashi Yamamuro

NTT Information and Communication Systems Laboratories

Feature-based similarity retrieval is a promising technique for retrieval of information-rich media data such as image, sound and video. We have been carrying out research on image retrieval systems called ExSight and HyperMatch that makes use of multiple features for similarity retrieval. Users can set arbitrary weight on each feature when querying. In this report, we propose two methods that can speed up retrieval in each feature by changing retrieval precision according to the weight on the feature, while keeping the final retrieval precision. We also carried out a preliminary experiment on one of the methods and the result showed the method's feasibility.

1. はじめに

パーソナルコンピューターとインターネット利用の普及により，ディジタルデータの量は，フロー／ストックとも，日に日に膨大になっている。これに伴い，これら大量のデータの中から必要なものを選別するための，検索技術への要求もますます大きくなっている。これは画像も例外ではなく，増えつづける画像データに対する有効な検索手段を模索する動きが始まっている。

われわれはこれまで，画像から自動抽出した特徴量をもとに検索を行うシステム ExSight + HyperMatch の開発を通じて，この問題の研究に取り組んできた [1, 2]。特徴量に基づく検索（CBR: Content-Based Retrieval, あるいは FBR: Feature-Based Retrieval）は，検索キーとして与えられた画像と，検索対象としてデータベース中に貯えられた画像の，それぞれから抽出された特徴量の類似度を計算・比較し，類似度の高いもの（すなわち「似ているもの」）を返すというかたちで実行される。検索に利用される特徴量は，画像から自動抽出できるため，検索対象となる画像データベースを構築する際の人手によるキーワード付与が不要で，そこから生じるさまざまな問題を回避できるという利点がある。

ExSight + HyperMatch では，ユーザーが画像の特徴（例えば色彩，形状など）を表す複数の特徴量について重みづけをして検索できる。特徴量に基づく類似検索においては，多次元木構造インデックスを利用した高速アクセス手法がよく知られているが，問い合わせごとに変化する重みづけに対応するために，検索エンジン HyperMatch では，個々の特徴量毎にインデックスを構築している。検索時には，まず個々の特徴量毎に類似度の高い画像を求めたのち，その結果を結合して全体の重みつき類似度を計算し，総合的に類似度の高い画像を返す仕組みになっている。

ここで問題になるのが，個々の特徴量において類似度の高い画像が，必ずしも与えられた重み条件において総合的に類似度の高い画像と一致しないという点である（以下，「全体順位／局所順位不一致問題」と呼ぶ）[2]。例えば，ユーザーがあるキー画像に類似した画像を10件返すように要求する問い合わせを行った場合，それぞれの特徴量において全て11番目の類似度を与える画像があったとすると，その画像が総合的な類似度においては1位となる可能性がある。

このため，HyperMatch では，個々の特徴量に対する要求回答数を，ユーザーの要求回答数より多くすることで，この問題から生じる検索精度の低下を緩和している。しかし，これはインデックスのノードへのアクセス回数を増加させ，検索速度の低下を招くことにもなる。

そこでわれわれは，これまで，指定された重みにかかわらずどの特徴量も一様に扱っていたのに対し，新たに，重みを考慮して個々の特徴量の検索を行う2つの手法を提案した。また，このうち1つの手法について小規模な実験を行い，フィージビリティを確認した。

2. 検索システム ExSight + HyperMatch

本稿で前提としている画像検索システム ExSight + HyperMatch の構成を図1に示す。画像データベースの構築および検索の主な流れは以下の通りである。なお，単純化のため，以後の記述では，本稿に関係のない ExSight, HyperMatch の機能についての説明は省略されている。

構築：
1) 画像からの特徴量抽出
2) 特徴量のデータベース投入
3) 多次元木構造インデックスの構築

検索：
4) 検索キーおよび検索条件の入力
5) 個々の特徴量の検索
6) 結果を統合→総合的な検索結果の返却

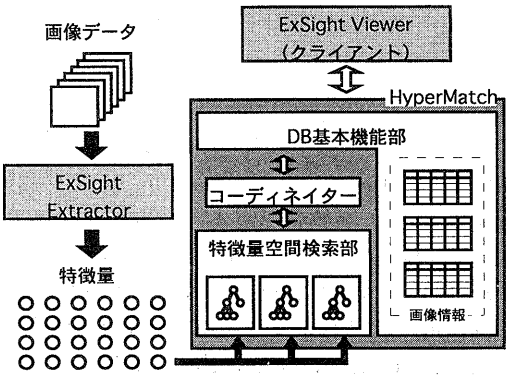

図1　システム構成

まず，構築過程について述べる。

最初に，与えられた画像から，その画像のさまざまな特徴を数値化して自動抽出する。この，特徴を数値化したものを「特徴量」という。特徴量は，一般に実数の組として表され，この組を，多次元空間中のベクトル（特徴量ベクトル）として扱うことが多い。例えば，色相の特徴量は16個の実数の組で表され，16次元ベクトルとして扱われる。抽出する特徴量は以下の8種類である。

色に関する特徴量（4種類）：
- ・色相
- ・彩度
- ・明度
- ・主観色基準

形に関する特徴量（2種類）：
- ・外周形状（[1]）
- ・ウェーブレット（[3]）

幾何情報の特徴量（2種類）：
- ・面積
- ・重心位置

こうして画像から自動抽出された特徴量ベクトルが，1画像あたり8ベクトルずつ，データベースに格納される。最後に，個々の特徴量ごとに多次元木構造インデックスを張って，データベースの構築が終了する。

ここで，それぞれの特徴量に一つずつインデックスを張るのではなく，全ての特徴量をひとまとめにした多次元空間に対して単一のインデックスを張ることもできる。しかし，この場合，検索時に各特徴量の重み（重視の度合）を変えることができない。したがって，われわれは前述の通り各特徴量ごとにインデックスをはる方法を採用している。

次に，検索過程について述べる。

まず，検索キーおよび検索条件が，以下のようなかたちで与えられる。

検索キー：

ユーザーは検索キーを画像として与える。インターフェイスモジュールは，この画像を上記8種類の特徴量ベクトルの組 $\{v_1, v_2, \ldots v_8\}$ に変換し，コーディネイターモジュールに引き渡す。

検索条件：

検索条件としては以下の3つがある。
- ・要求回答数 k

  類似度の高いものが上位何件まで必要かの指定。
- ・特徴量ごとの重み $\{w_1, w_2, \ldots w_8\}$

  その問い合わせにおいてそれぞれの特徴量をどれだけ重視するかの指定。0を最低，1を最高とする。この重みは，総合的な類似度を計算する際に利用される。計算方法については後述する。
- ・特徴量ごとの類似度評価関数の種類 $\{f_1, f_2, \ldots, f_8\}$

  ユーザーはまた，特徴量ごとに異なる類似度評価関数を指定できる。類似度評価関数についても後述する。

次に，個々の特徴量について検索を行う。重みが0の場合は，その特徴量については検索を行わない。

個別特徴量検索モジュールには，その特徴量における，

- ・キーベクトル v_i
- ・個別要求回答数 k_i
- ・類似度評価関数の種類 f_i

が，情報として渡される。キーベクトル v_i は，検索キーとして入力されたベクトルの組のうち，該当特徴量のベクトルである。要求回答数 k_i は，全体順位／局所順位不一致問題に由来する検索精度の低下を緩和するために，ユーザーの要求数 k に余裕係数 m（$m > 1$ なる定数）をかけたものとしている。したがって，$k_i = mk$ である。類似度評価関数の種類は，2つのベクトル間の類似度を数値化するための関数として何を使うかを指定する。現在我々のシステムで利用しているのは，L1（Manhattan 距離）関数，L2（Euclidean 距離）関数，A1（1次元非対称類似尺度）関数の3種類である。このうち A1 関数については木構造インデックスが利用できない [4]。

個別特徴量検索モジュールは，キーベクトル v_i とデータベース中の画像の該特徴量のベクトルとの類似度を，指定された類似度評価関数 f_i を用いて計算し，その結果をもとに，キーベクトルに類

似したものを上位から個別要求回答数 k_i 件ぶん返す。この際，現在利用可能な，どの類似度評価関数を用いても，実際に算出される値は「非類似度（dissimilarity)」，すなわち2つのベクトルが似ているほど小さく，似ていないほど大きく表される値となっている。また，このモジュールは，多次元木構造インデックスが利用できる場合，これを利用して類似度評価計算回数を減らし，検索速度の向上をはかる。

最後に，個々の特徴量検索の結果を統合して，最終的な検索結果を求める。コーディネイターモジュールは，個別特徴量検索モジュールから返された全ての画像について，与えられた重みに従って，総合非類似度 d を算出する。総合非類似度 d は，以下の式で与えられる。

$$d = \sum w_i d_i$$

d_i：i 番目の特徴量における非類似度

そして，d の小さいものから上位 k 件を最終検索結果として返す。

3. 重みを考慮したアクセスメソッド

前述したように，個別特徴量検索においては，多次元木構造インデックスの利用による速度向上がはかられている。多次元木構造インデックスでは，対象とする多次元空間中で，距離（＝非類似度）が近いベクトルを階層的にグループ化して木構造インデックスを構築する。この木構造を利用すると，検索時には，キーベクトルに近いベクトルに対してだけ類似度評価計算を実行すればよい。これにより，計算コストの大きい類似度評価計算の回数を削減して，検索速度を向上させる仕組みになっている。

例えば，ある検索を例に取ると，単一の特徴量空間（色相）について，96481件のベクトルの中から，わずか1545回の類似度評価計算で，キーベクトルに類似したもの上位70件を正確に返すことが可能である。インデックスがない場合，データベース中の96481件のベクトル全てについて類似度評価計算を行わなければならないことを考えると，木構造インデックスによる速度向上効果がかなりのものであることが理解できる。

このように，多次元木構造インデックスは，個別特徴量検索において非常に有効である。ところが，複数特徴量を組み合わせて検索を行おうとすると，新たな問題が浮上してくる。

複数特徴量の組み合わせ検索においては，個々の特徴量空間ごとに構築した多次元木構造インデックスを利用する。つまり，前節で述べたように，まず個々の特徴量からユーザー要求数より多い回答を求め，その候補群についてあらためて総合非類似度を計算し，最終検索結果を求めるという過程を踏むことになる。

ここで考慮しなければならないのは，個別特徴量における検索結果は，あくまで最終検索結果の候補にすぎないという点である。つまり，個別特徴量の検索結果と最終的な検索結果は必ずしも一致しない。これに，個別特徴量からユーザー要求数より多くの回答を取得することを考えあわせると，個別特徴量における検索は，必ずしも正確な上位 mk 件を返す必要がないといえる。

もちろん，個別特徴量における順位と最終順位のあいだには正の相関があるはずなので，個別特徴量の検索結果が全くでたらめであってよいわけではない。しかし，最終的な検索結果の精度を維持しつつ，個々の特徴量における検索精度をある程度犠牲にすることにより，一層の高速化をはかることは十分可能であると考えられる。では，どうすれば，最終的な検索精度を維持したまま個別特徴量における検索速度を向上することができるだろうか。

その手掛かりは特徴量ごとに与えられた重みにある。一般に，重みが大きい特徴量ほど，その特徴量における順位と最終結果の順位のあいだの相関は強いはずである。したがって，個別特徴量の検索を行う際に，重みの大きいものは従来通りの結果を返し，重みの小さいものについては精度を犠牲にして速度を向上させるようにすれば，最終結果の精度を維持したまま全体的な検索速度を向上させることが可能なはずである。

個別特徴量検索における精度を犠牲にして速度を向上させる方法としては，次の2つがあると考えられる。

一つは，余裕係数 m を小さくする方法である。m を小さくすると，その特徴量からの回答数が減る。このため，m を変化させる前に比べて，検索結果の情報量が減り，各特徴量からの候補集

合としては精度が落ちることになる。一方，必要な類似度評価計算回数も減るため，検索速度が向上する。

もう一つは，多次元木構造インデックスの検索アルゴリズムに近似係数 a を導入する方法である。多次元木構造インデックスにおける近似検索については，Arya らによって，近似係数を導入し，検索の正確さの下限を保証しつつ類似度計算回数を削減できることが証明されている [5]。したがって，与えられた重みに対応して a を変化させることで，個々の特徴量検索の精度を犠牲にして類似度計算回数を減らし，検索速度を向上させることができる。

以上から，各特徴量の重みを，m，a，あるいはこの両方に反映させることで，最終的な検索精度を維持したまま検索速度の向上を行えるのではないかと考えた。

4. 実験および考察

前節で，複数特徴量を利用する重み付き類似検索において最終的な検索精度を維持したまま検索速度の向上をはかる手法として，各特徴量の重みを，「余裕係数 m に反映させる」「近似係数 a に反映させる」の2つを挙げた。

このうち，後者の「近似係数 a に反映させる」について，フィージビリティを確認するための小規模な実験を行ったので，その結果について述べるとともに，考察を行う。

表1　各問い合わせにおける特徴量ごとの重み

問い合わせ	色相	彩度	明度	主観色基準	外周	ウェーブレット	面積	重心位置
1	1	0.2	0.1	1	0.1	-	1	0.3
2	0.5	0.5	-	0.8	0.3	-	1	0.2
3	1	0.6	0.1	0.5	0.2	-	1	1
4	0.9	0.2	0.4	1	0.8	-	-	-
5	0.9	0.5	-	0.9	0.5	-	0.5	-
6	0.5	0.5	-	-	0.5	-	-	-
7	0.5	0.5	-	0.5	0.2	0.2	-	-
8	1	0.5	0.2	-	1	0.5	-	-
9	0.9	0.3	0.1	0.8	0.1	-	0.6	0.3
10	0.1	-	-	0.2	1	0.3	0.9	0.6
11	0.7	0.2	-	0.6	0.5	-	-	-
12	1	0.5	-	1	0.5	-	0.9	0.8

注：　-　は重み0で検索されないことを示す

4.1. 実験条件

実験は，Sun 社の Ultra 30 ワークステーション（OS: Solaris 2.5.1）上で行った。

画像データとしては，写真画像1009枚から物体抽出 [1] を用いて切り出した96481枚の子画像を対象とした。この子画像から2節で述べた8つの特徴量を抽出し，検索エンジン HyperMatch に投入したのち，各特徴量について多次元木構造インデックスを構築した。多次元木構造インデックスとしては，ノードサイズ 200 の VAMSplit R-tree [6] を用いた。

問い合わせは，ExSight + HyperMatch システムで実際に画像検索に使用された問い合わせの中から，代表的なもの12例を使用した（表1）。但し，単純化のため，総ての問い合わせで，要求回答数 k は70に，類似度評価関数 f は L1 に固定した。この結果，全ての問い合わせの全ての個別特徴量検索においてインデックスが利用された。

検索時のパラメータである余裕係数 m および近似係数 a については，以下のように設定した。まず，余裕係数は m は固定した。実際は，[2] に基づき $m = 6$ の値を用いた。一方，近似係数 a については，2通りの手法で実験を行った。

一つは，重みに関係なく，どの特徴量でも $a = 1$ とする手法（以下「近似なし手法」と呼ぶ）である。これは個別特徴量検索モジュールが正確に非類似度の小さいほうから mk 件の回答を返す設定で，従来の HyperMatch で採ってきた方法である。

もう一つは，個別特徴量の検索において，該特徴量の重み w_i ($0 \leq w_i \leq 1$) をそのまま近似係数に代入し，近似係数 $a_i = w_i$ で個別特徴量の検索を行う手法（以下「重み反映近似手法」である。この2つの異なる手法で得られた結果を比較して，重みを近似係数に反映させる手法のフィージビリティを評価した。

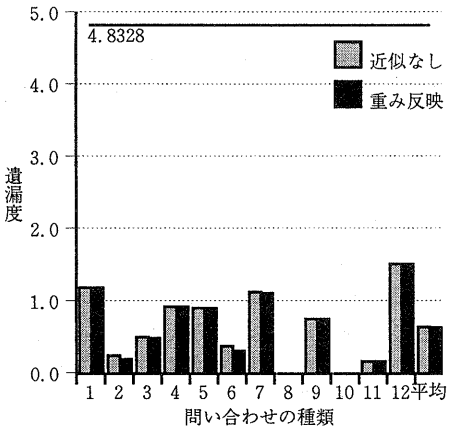

図2　遺漏度

また，これとは別に，各試験問い合わせについて全件計算を行い，重み付き総合類似度上位70件の正解集合を得た。

4.2. 実験結果および考察

まず，2つの手法について，各問い合わせごとに遺漏度を算出した結果を，図2に示す。

遺漏度は，k-近件検索の正確さを評価するために考案した指標で，正解集合と回答集合を比較し，正解集合の順位 r 位の要素が回答集合から漏れていた場合，遺漏度には $1/r$ の値が加算される。つまり，遺漏度 o は，

$$o = \sum 1/r_i$$

r_i：回答集合から漏れていた要素の正解集合における順位

で表される。この指標は，同じ1件の検索漏れであっても，1位が漏れていた場合と70位が漏れていた場合では，1位が漏れているほうがより深刻であることを評価に反映させるために考案された。この定義から分かるように，遺漏度は小さいほどよく，遺漏度0で正解と完全に一致していることを意味する。

図2から分かるように，12問い合わせの遺漏度の平均は，近似なし手法では0.6445，重み反映近似手法では0.6332と，ほとんど差がなかった。また，個々の問い合わせごとに比べても，やはり手法による違いはほとんど見られない。

つまり，重み反映近似手法では，個々の特徴量における検索精度が低下しているにもかかわらず，全特徴量を統合した検索結果では，検索精度にほとんど差がない。また，この傾向は問い合わせの内容にも左右されていない。

参考までに，70-近件検索における遺漏度は最大（すなわち70件全ての回答が間違っていた場合）で4.8328である。一方，今回の実験では，問い合わせごとの遺漏度の違いがけっこう大きく，最良で0，最悪で1.5強であった。このことは，今回の実験では固定であった余裕係数 m を，条件に応じて大きくしたり小さくしたりする必要があることを示しているのではないかと考えている。

図3は，各問い合わせごとに，類似度評価計算回数の総和を重みが0でない（すなわち実際に検索された）特徴量の数で割った値（以下，「特徴量あたりの計算回数」と呼ぶ）を示したものである。参考までに，インデックスを利用しなかった場合，特徴量あたりの計算回数は全画像データ数，すなわち96481回になる。

この図から分かるように，重み反映近似手法における特徴量あたりの計算回数は，近似なし手法に比べて大幅に減少しており，全問い合わせの平均では約56％の減少，つまり半分以下に減っている。また，個々の問い合わせを見ても，全ての問い合わせにおいて特徴量あたりの計算回数が減少

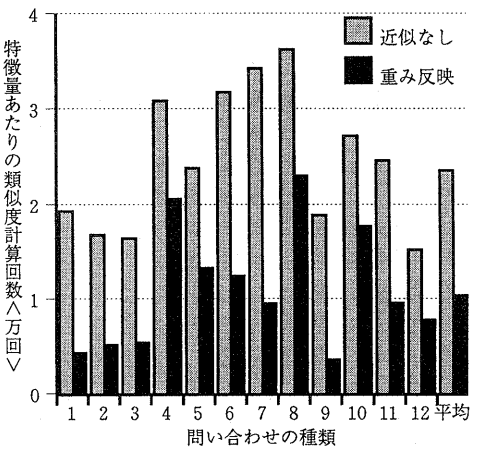

図3　特徴量あたりの計算回数

していて，最低でも約33%，最高では約80%の計算回数の削減を実現している。

つまり，重み反映近似手法は，特徴量あたりの計算回数の削減に顕著な効果があるといえる。ただし，その効果の程度は，問い合わせの内容に依存する。しかし，ここで用いた試験問い合わせが，いずれも実際の画像検索において使用されたものであることを考えると，程度に差こそあれ，実用場面における重み反映近似手法にはかなりの効果があるといえるのではないかと考える。

以上2つの結果から，重み反映近似手法は，最終的な検索精度を低下させることなく，検索速度の向上させるという点において，非常に有望な手法であることが示されたといえる。

5. まとめ

複数特徴量の重み付き類似検索において，最終的な検索精度を低下させることなく，個々の特徴量の検索精度を犠牲にして検索速度を向上させる手段として，特徴量ごとの重みを余裕係数および近似係数に反映させる2つの手法を提案した。このうち，特徴量ごとの重みを近似係数に反映させる手法について，実際の検索に利用された問い合わせを用いて小規模な実験を行った。この実験結果から，この手法が最終的な検索精度を低下させることなく検索速度を向上させる手段として非常に高いフィージビリティをもつことを示した。

今後は，主に次の4つの方針で更なる研究を行っていく予定である。

- 余裕係数 m に重みを反映させる手法，および2手法の組み合わせの評価

  今回実験を行わなかった余裕係数を重みに反映させる手法，および，2手法を組み合わせた場合についても，評価実験を行う。

- 余裕係数 m, 近似係数 a への重みの反映させかたの検討

  今回の実験では，近似係数と重みの関係式を，$a_i = w_i$ とした。しかし，最終的な検索精度を低下させることなく，可能なかぎり計算回数の削減を追求するために，例えば $a_i = w_i^2$ のような式を用いた場合の評価も行いたい。同様の検討は，重みから余裕係数を導出する式についても必要である。

- 異なる多次元木構造インデックス C-tree [7] についての評価

  われわれの開発した多次元木構造インデックス C-tree を利用した場合，どのような違いが現れるかについて，評価実験を行う。

- 実験規模の拡大

  より大規模な実験によって，手法の妥当性を確認する。

以上を通して得られた成果は，今後の ExSight + HyperMatch 検索システムに随時取り込んでいく予定である。

参考文献

[1] 赤間, 紺谷, 三井, 串間, "画像内オブジェクトの自動抽出を使った画像検索システム - ExSight -, " 信学会, 第8回データ工学ワークショップ, 1997.

[2] Curtis, Taniguchi, Nakagawa, and Yamamuro, "A Comprehensive Image Similarity Retrieval System that utilizes Multiple Feature Vectors in High Dimensional Space," 情処学会 DBS-113, 1997.

[3] Konya, Kushima, "A Rotation Invariant Shape Representation Based Wavelet Trasform," British Computer Society 'The Challenge of Image Retrieval' Workshop, 1998.

[4] Taniguchi, Yamamuro, "Multiple Inverted Array Structure for Similar Image Retrieval," The Proceedings of International Conference on Multimedia Computing Systems '98 (ICMCS98), 1998 (To appear).

[5] Arya, Mount, Netanyahu, Silverman, Wu, "An Optimal Algorithm for Apprximate Nearest Neighbor Searching in Fixed Dimensions," The Proceedings of Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, 1994.

[6] White, Jain, "Similairty Indexing: Algorithms and Performance," The Proceedings of SPIE: Storage and Retrieval for Image and Video Databases IV, 1996.

[7] Curtis, Nakagawa, Taniguchi, Yamamuro, "Similarity Indexing in High Dimensional Image Space," 情処学会 DPS-82, 1997.