画像の類似検索に向けた多次元インデクス手法

上川伸彦† 岩崎一正†

近年，画像に対する高速類似検索への要求が高まっている．画像の類似検索は画像としての特徴を表す多次元ベクトルの距離計算で実現されることが多いため，多次元ベクトルに対するインデクス手法が注目されている．従来の多次元インデクス手法といえるツリー構造の多次元検索手法は，適用可能であるベクトルはせいぜい数次元程度である．しかし，画像の類似検索においては，特徴を表す多次元ベクトルは数十次元ないし数百次元に及ぶ．そこで，本論文では，画像の類似検索の特性に着目し，数十次元ないし数百次元のベクトルを対象にしても高速に類似検索を行うことができる多次元インデクス手法を報告する．

High-dimensional Indexing Methods for Similarity-search of Image Contents

Nobuhiko Kamikawa† and Kazumasa Iwasaki†

Recently, requirement of fast similarity-search for images is increasing. The indexing method for n-dimensional vector has great interest, because methods of the similarity-search for images usually use n-dimensional vectors have features of the image to calculate similarity. A feature vector of an image often has dimensions over a hundred but precedent method of the similarity-search are not work effectively for High-dimensional vectors. In this paper we propose a method of the similarity-search who can work effectively for High-dimensional vectors.

1. はじめに

マルチメディア・データのデジタル化にともない，従来のテキストベースのデータ検索技術以上に，マルチメディア・データを検索する技術について広く関心が集まっている．そして近年，マルチメディア・データの1つである画像の類似検索手法が実用化されつつある．検索処理は，1枚の画像から，画像としての特徴を表す数十次元ないし数百次元に及ぶ多次元ベクトル（以下，特徴量ベクトルと呼ぶ）を算出し，特徴量ベクトルどうしの距離を計算することにより，類似する画像を検索する手法が一般的である．このような類似検索手法は，検索対象となる画像1枚ごとに特徴量ベクトルどうしの演算を行う必要があるため，検索処理に必要な時間が検索対象となる画像件数に比例して増加してしまう．そのため，検索処理の高速化を行わないと実用化し難い面がある．そこで，従来から行われているツリー構造の多次元インデクス手法を数十次元ないし数百次元に及ぶ特徴量ベクトルに適用する研究・開発が行われている[1)~3)]．一方で，従来行われているツリー構造の多次元インデクス検索手法では，対象ベクトルが10次元以上になると検索を高速化することはできないといわれている[4)]．

また，システム環境の面から見ると，ネットワークの高速化や記憶装置の大容量化にともない，大量の画像を管理する要求が高まっている．従来，このような画像を管理するための検索としては，テキストベースの検索手法が主流であった．すなわち，画像の蓄積時に，人間がテキスト情報を付加して元の画像と関連付けて蓄積しておき，検索時に事前に付加されたテキストデータに基づいて検索を行う，という手法である．しかし，蓄積する画像の件数が増大すると，画像にテキスト情報を付加する作業のコストも増大してしまう．また，上述したように画像の特徴によって検索する手法は実用化されつつあり，従来のテキストベースの検索手法と併用して画像を管理するようになるのは想像に難くない．今後，画像に対する高速な類似検索手法が必要となるのは明らかである．そこで本論文では，画像を対象とした類似検索の特性を考慮して，数十次元ないし数百次元の特徴量ベクトルを用いる画像の類似検索を高速化する手法について報告する．

† 日立製作所ビジネスソリューション事業部
Business Solution Systems Division, Hitachi, Ltd.

2. 類似画像検索の高速化

ここでは，類似画像検索における課題と従来の多次元ベクトル検索の高速化手法について述べ，本論文における類似画像検索を高速化するためのアプローチについて説明する．

2.1 類似画像検索における課題

類似画像検索とは，検索者が指定する検索キー画像に類似した画像を検索する技術である．従来のテキストの一致検索と違って，検索キー画像と一致する画像を検索するわけではないので，「類似」の概念を計算機上でどう表現するかが重要である．そこで，まず 1 枚の画像から画像の特徴を表す「特徴量ベクトル」を算出し，検索処理において，ベクトル二乗距離等の特徴量ベクトルどうしの演算結果を類似度として用いることにより，類似判定を行う手法が一般的となっている（**図 1**）．この特徴量ベクトルは，たとえば，画像の色情報やテクスチャ情報等多くの情報を持ち，数十次元ないし数百次元に及ぶことが多い．

このように，一般的な類似画像検索手法は特徴量ベクトルどうしの演算処理を画像 1 枚ごとに行う必要があるため，検索処理コストが画像枚数に比例する．よって，大量の画像を検索対象とした場合に，検索処理の応答時間が非常に長くなってしまうという問題がある．そのため，類似画像検索の普及にともない，類似画像検索の応答時間を短縮する手法への要求が高まっている．

2.2 多次元ベクトル検索の高速化手法

(1) ツリー構造の検索手法

これまで，多次元ベクトルを対象にした検索高速化手法として，SR-Tree [1] や X-tree [2]，UB-Tree [3] といったような，様々なツリー構造の多次元インデクス手法が提案されている．これらの手法は地図データに代表されるような，数次元のベクトルを対象に研究・開発されたものであるといえる．地図データにおける，たとえば「検索キーの多次元ベクトルが示す地点から一番近い場所を検索する」といったような要求に応える検索手法である．このような，低次元ベクトルを対象に研究・開発された従来のインデクス手法を，数十次元ないし数百次元の特徴量ベクトルを扱う類似画像検索にそのまま適用するには，いまだ解決されていない多くの課題がある．その課題は，多次元空間が高次元化すると非常に広大な空間になる，という現象に起因している．すなわち，多次元空間の体積は次元数の累乗に比例するということであり，多次元空間をいくつかの領域に分割した場合に，1 つの領域に隣接する領域の数が増大するということである．

(2) 非ツリー構造の検索手法

最近になって，数十次元にも及ぶ多次元ベクトルを対象とした類似検索においては，従来のツリー構造の検索手法では高速化できないことが分かってきた．すなわち，10 次元以上のベクトルを対象にした場合，ツリー構造の検索手法よりも全件走査の方が高速に類似検索を行えるのである．そのため，VA-File 法[4] や転置ファイル法[5] といったような，ツリー構造を持たない多次元ベクトルの検索高速化手法が開発されている．しかし，どちらの検索手法においても，以下のような課題が残されており，今すぐ実用化できるわけではない．

- VA-File 法[4]
  - 検索性能がデータ件数に比例する
  - 類似検索への適用に対する検討が弱い
- 転置ファイル法[5]
  - 検索精度が悪化する場合がある
  - データの登録に時間がかかる
  - ディスク使用量が多い

2.3 類似画像検索高速化へのアプローチ

本論文では，検索において数十次元ないし数百次元もの高次元ベクトルを扱う多くは，マルチメディア・データ，現状ではほとんど画像，の類似検索であることに着目し，類似画像検索の高速化に特化した多次元インデクス手法を提案する．検索機能として，特徴量ベクトル空間でのベクトル距離ではなく，特徴量ベクトルの各次元値を基準にして検索結果を求める．すなわち，従来の特徴量ベクトル検索手法は検索範囲が超球形であり，特徴量ベクトル空間で検索キーとなる特徴量ベクトルを中心とする超球形領域の内部に存在する特徴量ベクトルの集合を検索結果とするのに対して，提案手法では，検索範囲を超矩形領域として，特徴量

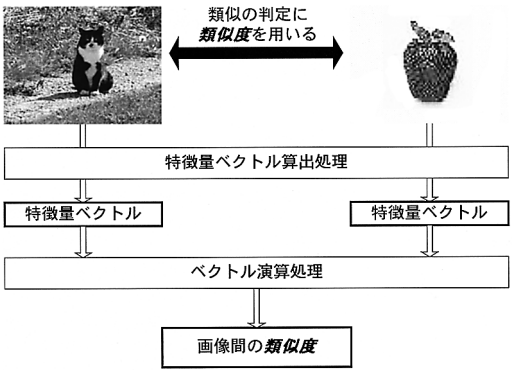

図 1 類似画像検索
Fig. 1 Similarity-search of image contents.

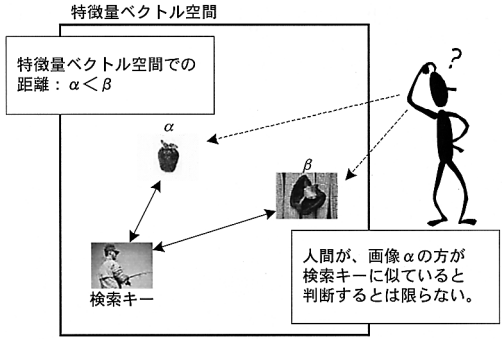

図 2 画像の類似度
Fig. 2 Similarity of image.

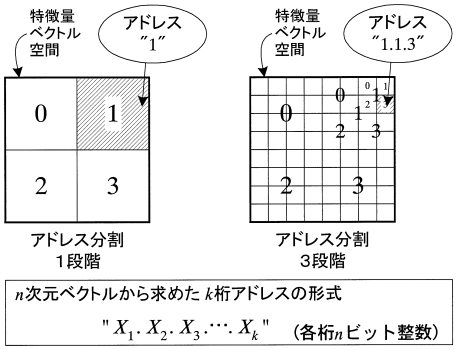

図 3 アドレス方式
Fig. 3 Addressing form.

ベクトル空間内で検索キーとなる特徴量ベクトルを含む超矩形領域の内部に存在する特徴量ベクトルの集合を検索結果とする．

これにより，以降で提案するように，数十次元以上のベクトルデータを対象とした類似検索の高速化が可能となる．また，検索範囲の形状を変化させても検索精度が著しく悪化することはない．これは，類似画像検索においては，地図データのような従来のベクトル検索とは違い，「類似」の定義が明確ではなく，特徴量ベクトル空間内での距離に基づいた類似度が必ずしも人間の判断と一致するとは限らない（**図 2**）からである．

3. 類似画像検索に向けた多次元インデクス手法

本論文で提案する多次元インデクス手法では，画像から算出される数十次元ないし数百次元の特徴量ベクトルを 1 次元的な順序を持つ「アドレス値」で近似する．検索対象である特徴量ベクトルが検索範囲の内部に存在するか否かを，アドレス値を用いて粗く判定することによって，検索結果の候補となる特徴量ベクトルを絞り込む．そして絞り込まれた特徴量ベクトルに対してのみ，特徴量ベクトルを用いて検索範囲内に存在するか否かの判定を行う．このように，数十次元ないし数百次元の特徴量ベクトルの計算処理をアドレス値による計算処理とごく少量の特徴量ベクトルの計算処理で代替することにより，類似検索処理にかかわる計算量を減らすことが可能となる．またアドレス値は，B-Tree アルゴリズムを適用して類似画像検索処理を高速化するように 1 次元的な順序付けがされているので，さらに類似検索処理にかかわる計算量を減らすことができる．

以上のように，本論文の多次元インデクス手法では，「特徴量ベクトルのアドレス値への近似」と「アドレス値への B-Tree アルゴリズムの適用」によって類似画像検索処理にかかわる計算量を減らし，結果として類似画像検索処理を高速化している．以下，アドレス値の算出方法と，アドレス値を用いた類似画像検索処理の流れについて説明する．

3.1 アドレス方式

アドレス値は，特徴量ベクトルから算出される k 個の数値並びで，これを k 桁アドレス値と呼ぶ．アドレス値の桁数 k はアドレス分割の段階数を表す．アドレス分割とは，1 個の n 次元超矩形領域を各次元軸 2 等分割し，2^n 個の超矩形領域に分割する操作である．1 段階目のアドレス分割では特徴量ベクトル空間全体を 1 個の超矩形領域として分割を行い，2 段階目のアドレス分割では 1 段階目のアドレス分割で分割されてできた超矩形領域をそれぞれ分割する，といったように階層的に分割を行う．最終的に，k 段階のアドレス分割で特徴量ベクトル空間が 2^{kn} 個の超矩形領域（以下，アドレス領域と呼ぶ）に分割され，2^{kn} 個の各アドレス領域を識別できる数値として，k 桁の n ビット整数がアドレス値として各アドレス領域に割り当てられる（**図 3**）．つまり，特徴量ベクトルをアドレス値で近似するということは，特徴量ベクトル空間を分割してできた 2^{kn} 個のアドレス領域のうち，特徴量ベクトルがどのアドレス領域に含まれるかをアドレス値で表すということである．

ベクトルの各次元値が float 値（4 Byte）だとすると，32 次元の特徴量ベクトル（128 Byte）から 2 桁アドレス値（8 Byte）を算出する場合には 1/16 に，128 次元の特徴量ベクトル（512 Byte）から 4 桁アドレス値（64 Byte）を算出する場合には 1/8 に特徴量ベクトルの情報を圧縮することになる．また，アドレス値を算出する際に，特徴量ベクトルの全次元値を使用せず，ある程度の次元を間引いたベクトルからアド

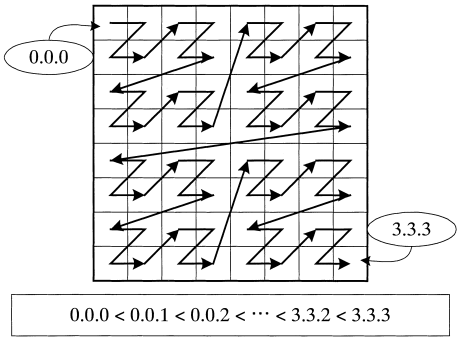

図 4 アドレス値の順序
Fig. 4 Order of the address.

レス値を算出することによって，さらに圧縮率を高めることができる．たとえば，128 次元の特徴量ベクトル（512 Byte）から次元を間引いて 32 次元ベクトル（128 Byte）を作成し，作成した 32 次元のベクトルから 4 桁アドレス値（16 Byte）を算出すれば，1/32 に特徴量ベクトルの情報を圧縮することができる．

また，本論文で提案する多次元インデクス手法では，アドレス値に**図 4** のような順序付けの規則を設ける．すなわち，アドレス値の最上位桁の数値が小さいアドレス値ほど小さいと判断し，上位桁の数値が同じ場合にはその次の桁の数値で判断する．このような順序付けを持つので，特徴量ベクトルをアドレス値で近似すれば，特徴量ベクトルに対してアドレス値をキーとして B-Tree アルゴリズムを適用することができる．

以上述べたアドレス値の規則は，UB-Tree[3]において，UB-Tree を格納するディスクのページ境界を表すのに用いられているのと同様であるが，本論文の提案手法においては，(1) アドレス値の桁数を固定にする，(2) アドレス値のサイズをなるべく小さくするといった変更を加えた．

3.2 類似検索処理

本論文で提案する類似検索手法は，特徴量ベクトル空間内において，検索者が指定した検索範囲内に含まれる特徴量ベクトルを求める機能を持つ．ここで検索範囲とは，n 次元の特徴量ベクトルを検索対象とした場合，各次元の辺が $2\varepsilon_i$ $(i = 1, 2 \cdots n)$ であり，検索キーとなる特徴量ベクトルを包含する n 次元の超矩形領域とする（**図 5**）．この検索範囲は，検索条件として厳しくする属性に対応する次元軸辺を小さく，緩やかにする属性に対応する次元軸辺を大きくして指定すればよい．

検索処理の流れは，(1) アドレス値による候補絞り込み，(2) 特徴量ベクトルによる結果判定の 2 段階の処理により検索結果となる特徴量ベクトルの集合を求

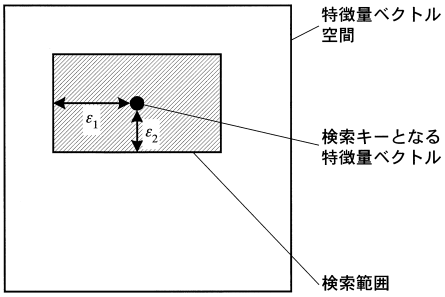

図 5 検索範囲
Fig. 5 Query range.

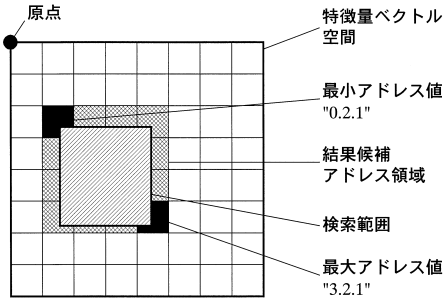

図 6 結果候補アドレス領域
Fig. 6 Address area with the candidate.

める．以降，それぞれの処理について説明する．

(1) アドレス値による候補絞り込み

アドレス値による候補絞り込みの処理では，検索範囲と重なるアドレス領域に含まれる特徴量ベクトルを求める．このアドレス値による候補絞り込み処理は，(a) 結果候補アドレス領域を求める，(b) 候補絞り込み処理の開始位置を求める，(c) アドレス値よる候補絞り込み処理を行うという 3 段階の処理に分けることができる．

(a) 結果候補アドレス領域を求める

結果候補アドレス領域とは，検索範囲と重なるアドレス領域の集合で構成される超矩形領域であり，検索範囲における最小アドレス値と最大アドレス値とを算出すれば求めることができる．検索範囲における最小アドレス値とは，検索範囲において特徴量ベクトル空間の原点から最も近い点のアドレス値であり，検索範囲における最大アドレス値とは，同様に検索範囲において特徴量ベクトル空間の原点から最も遠い点のアドレス値である．この検索範囲における最小アドレス値と検索範囲における最大アドレス値とを対角線とする超矩形領域が結果候補アドレス領域である（**図 6**）．

(b) 候補絞り込み処理の開始位置，終了位置を求める

ここでは，結果候補アドレス領域を構成する各アドレス領域のアドレス値は，**図 7** からも分かるように，

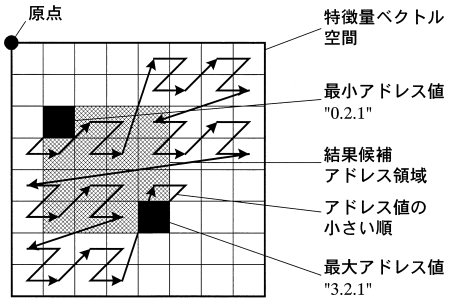

図 7 検索範囲と最小アドレス値，最大アドレス値
Fig. 7 Query range and Maximum address, Minimum address.

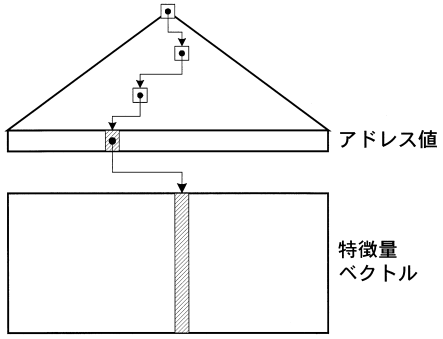

図 8 アドレス値をキーとした B-Tree
Fig. 8 Construction of B-Tree with address.

検索範囲における最小アドレス値以上であり，検索範囲における最大アドレス値以下である．そのため，検索範囲における最小アドレス値よりも小さいアドレス値を持つ特徴量ベクトルと，検索範囲における最大アドレス値よりも大きいアドレス値を持つ特徴量ベクトルに対しては，アドレス値による候補絞り込み処理を行わなくても，検索範囲の内部には存在しえないと判定できる．よって，候補絞り込み処理の開始位置を検索範囲における最小アドレス値の蓄積位置とし，候補絞り込み処理の終了位置を検索範囲における最大アドレス値の蓄積位置とし，候補絞り込み処理の開始位置からアドレス値の小さい順に候補絞り込み処理を行うこととする．ここで，本論文で提案する検索手法では，画像の特徴量ベクトルを蓄積する際に，特徴量ベクトルからアドレス値を算出し，**図 8** のようなアドレス値をキーとした B-Tree を作成しておくので，候補絞り込み処理の開始位置を容易に求めることができる．

(c) アドレス値による候補絞り込み処理を行う

ここでは，候補絞り込み処理の開始位置に蓄積されているアドレス値から，候補絞り込み処理の終了位置に蓄積されているアドレス値まで，アドレス値の小さい順にアドレス値による候補絞り込み処理を行う．絞り込み処理は，**図 9** に示す，アドレス値を各次元軸に分割した値である軸アドレスを使用する．検索範囲における最小アドレス値を $amin$，検索範囲における最大アドレス値を $amax$，検索対象の特徴量ベクトルが持つアドレス値を a とすると，次式を満たすアドレス値 a を持つ特徴量ベクトルが結果候補アドレス領域に含まれると判定される．なお，a_i とはアドレス値 a の i 軸アドレス，$amin_i$ とはアドレス値 $amin$ の i 軸アドレス，$amax_i$ とはアドレス値 $amax$ の i 軸アドレス，k とはアドレス値 a の桁数を表す．

$$amin_i < a_i < amax_i \quad (i = 1, 2, \cdots, k) \qquad (1)$$

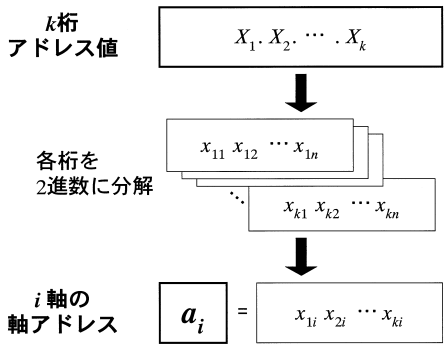

図 9 軸アドレス
Fig. 9 Axi-address.

(2) 特徴量ベクトルによる結果判定

ここでは，アドレス値による候補絞り込みの処理で結果候補アドレス領域に含まれると判定された特徴量ベクトルに対して，特徴量ベクトルが検索範囲内に存在するか否かを，特徴量ベクトルを使って判定する．判定は，特徴量ベクトルの各次元値と検索範囲の各次元値とを次元ごとに比較して行う．つまり，検索キーとなる特徴量ベクトルを key とすると，次式を満たす特徴量ベクトル x が検索範囲内に含まれると判定される．なお，x_i とは，特徴量ベクトル x の i 次元軸の値，n とは特徴量ベクトルの次元数を表す．

$$key_i - \varepsilon_i < x_i < key_i + \varepsilon_i \quad (i = 1, 2, \cdots, n) \qquad (2)$$

特徴量ベクトルによる結果判定処理によって，検索範囲内に含まれると判定された特徴量ベクトルが，検索キーとなる特徴量ベクトルに類似した特徴量ベクトルを持つ画像，すなわち，検索結果として求められる．

ここで，図 8 に示すように，特徴量ベクトルとアドレス値は別々に蓄積されているので，I/O コストについていうと，アドレス値による候補絞り込み処理では蓄積されているアドレス値に対するシーケンシャルなファイルアクセスを行うに過ぎず，特徴量ベクトルに

対するファイルアクセスを行うのは，アドレス値による候補絞り込み処理で結果候補だと判定された特徴量ベクトルに対してのみである．

4. 検索性能試験

4.1 使用したデータ

ここまで説明した多次元インデクス手法を PC サーバ上に実装して，類似画像検索機能の性能試験を行った．使用したデータは，市販されている画像素材集から無作為に選択した 10 万枚のカラー写真である．特徴量ベクトルには，色情報や輝度情報から算出した 59 次元のベクトルと 531 次元のベクトルを使用した．ここで，提案手法の検索性能はベクトルデータの多次元空間内での分布によって変化する．そのため，性能試験では，提案手法が対象とするデータ，すなわち一般的な類似画像検索で使用されると考えられるデータを用いた．用いたデータは多次元空間内で一様に分布しているわけではなく，部分的な粗密の偏りがある．

特徴量ベクトルからアドレス値を算出する際には，特徴量ベクトルから次元を間引いて 32 次元のベクトルを作成し，その 32 次元のベクトルからアドレス値を算出した．すなわち，59 次元の特徴量ベクトルを使用した場合は，59 次元の float 値（4 Byte）を持つ特徴量ベクトル（236 Byte）を 32 次元の 6 桁アドレス値（24 Byte）に，531 次元の特徴量ベクトルを使用した場合は，531 次元の float 値（4 Byte）を持つ特徴量ベクトル（2124 Byte）を 32 次元の 6 桁アドレス値（24 Byte）に圧縮して，類似検索処理の高速化を図っている．

実装方法は，まず画像を蓄積する際に，画像 1 件ごとに，画像から特徴量ベクトルを算出し特徴量ベクトルを蓄積する．同時に，特徴量ベクトルからアドレス値を算出し，アドレス値をキーとした B-Tree を作成して蓄積する．このアドレス値をキーとした B-Tree と蓄積した特徴量ベクトルが，本論文で提案する多次元インデクス手法において使用するデータであり，今回は WindowsNT の OS ファイルとして作成した．

4.2 応答時間

ここでは，

(1) 1 件ごとに特徴量ベクトルを用いて計算を行う全件走査方式（full scan）

(2) 本論文で提案する多次元インデクス（index）

の 2 方式に関して，59 次元の特徴量ベクトルと 531 次元の特徴量ベクトルのそれぞれに対して類似検索を行ったときの応答時間を測定した．方式 (1) とは，10 次元以上のベクトルを対象とした類似検索において，

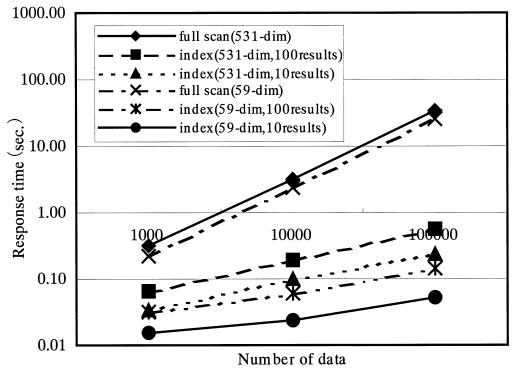

図 10 結果 p 件となる類似検索の応答時間
Fig. 10 Response of range query for p-results.

超球形の検索範囲を用いた場合の最速の検索方式である．方式 (2) については，検索処理コストが検索結果件数に依存するので，結果件数が 10 件となるような検索範囲（10-results）と検索結果が 100 件となるような検索範囲（100-results）に対して類似検索を行ったときの応答時間を測定した．また，方式 (2) における応答時間は検索キー付近の特徴量ベクトルの分布にも影響を受けるので，10 種類の検索キーに対する測定の平均値を示した．なお，蓄積画像数は 1000 件，1 万件，10 万件と変化させて測定した．測定結果を**図 10** に示す．

測定結果から分かるように，全件走査方式ではデータ件数に比例して応答時間が増加しているが，本提案手法ではデータ件数による応答時間への影響が少ないことが分かる．590 次元もの高次元ベクトルを対象とした類似検索においても，本提案手法の応答時間は全件走査方式の応答時間に対して，データ件数 1000 件の場合で 1/10 以下に，データ件数 10 万件の場合で 1/200 以下に短縮されている．

また，検索結果件数が 10 件から 100 件に増加した場合に，応答時間が 3 倍程度に増加しているのが分かる．すなわち，検索者が検索範囲を変更して再検索を行った場合に，検索結果件数が 10 倍になる場合には 3 倍程度の応答時間を必要とすることを意味する．

4.3 検索精度

まず，多次元空間内でのベクトル二乗距離を基準とした精度について述べる．提案手法による検索結果は，検索キーベクトルからの距離が近い順に集めたベクトルの集合ではない．これは，提案手法で使用する検索範囲の形状が，超球形ではなく超矩形領域であることに起因する．そのため，距離の近い順に集めたベクトル集合を検索結果とするべき類似検索に適用する場合，検索精度が悪化するのは想像に難くない．

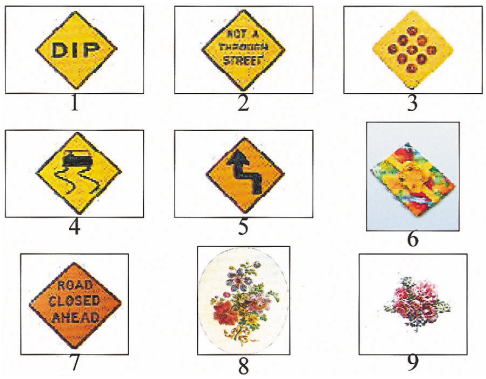

図 11 多次元空間内での距離順の検索結果
Fig. 11 Result of full scan search.

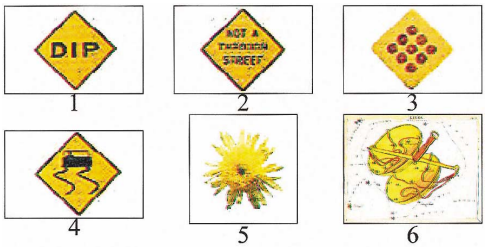

図 12 提案手法での検索結果
Fig. 12 Results of our index search.

次に，提案手法が対象としている一般的な類似画像検索での「見た目の精度」について考察する．**図 11** に多次元空間内での距離順に類似画像検索を行った結果上位 9 件を，**図 12** に提案手法で類似検索を行った結果 6 件を距離順に並べ替えた結果を示す．どちらの場合も，1 番の画像を検索キーとして，距離の近い順に 1 番から番号をふってある．なお，使用した特徴量ベクトルは，色と形状が類似した画像を検索するものである．図 11 と図 12 を比較して分かるとおり，明らかに色と形が似ていると思われる画像（図 11 の 1，2，3，4）は提案手法において検索結果となっている．また，形がよく似ていると思われる画像（図 11 の 5，7）に関しては，色が似ていないために提案手法においては検索結果とはなっていない．ここでは検索結果の一例のみを用いて説明したが，他の画像を検索キーとした場合でも，明らかに似ている画像が検索結果となる傾向に変わりはない．

以上述べたように，一般的な類似画像検索での「見た目の精度」に関して，提案手法による検索結果は距離を基準とした検索結果に比べて，著しく悪化することのないことが分かった．

5. 考　　察

本論文で提案した手法の特長を以下に述べる．

(1) 高速な類似画像検索

提案手法では，10 次元以上のベクトルに対して，最速の検索方式である全件走査方式の検索手法に比べ，10 万件の画像に対する検索で 1/200 の応答時間を実現している．これは，現在構築されている画像データベースにおける類似画像検索の応答時間を短縮できることを意味する．言い換えると，さらに大量の画像を管理するデータベースを構築しても，実用的な応答時間の類似画像検索を提供することができる．

(2) 広い適用範囲

今後，多種多様なマルチメディア・データをデジタルとして大量に管理することが一般的になると考えられる．提案手法では，様々な特徴量を持つマルチメディア・データに対応できるため，類似画像検索だけではなく，現在活発に研究されている音楽や映像等の類似検索や，特徴量ベクトルを算出できるその他多くのマルチメディア・データの類似検索に容易に実装することができる．

(3) 比較的容易な実装

提案手法では，アドレス値をキーとした B-Tree アルゴリズムを使用している．すなわち，現在，1 次元データを対象にした一般的な高速検索手法として定着している B-Tree アルゴリズムの既存資源を使用することにより，データのメンテナンス処理や検索処理の一部を容易に実装することができる．

一方，提案手法には，検索者にとって使いやすい類似画像検索を実現するためにはいくつかの課題がある．提案手法は特徴量ベクトル空間で任意の超矩形領域内に存在する特徴量ベクトルを検索する機能を持つ．つまり，検索者が指定するのは検索範囲となる超矩形領域であり，検索結果件数ではない．そのため，指定した検索範囲の内部に特徴量ベクトルが存在しない，すなわち検索結果が 0 件となることがある．マルチメディア・データの類似検索においては，検索キーとなるマルチメディア・データに似ている順番に数件のデータを即時に表示・再生することを要求される場合が多い．そのため，検索結果 0 件の場合には，検索範囲を少し広げて検索処理を繰り返す必要がある．このような検索の繰返しを行うと，当然，検索時間が長くなってしまう．また，このような検索処理の繰返しを避けるために，最初の検索実行時に広めの検索範囲を指定すると，逆に検索結果件数が増えすぎてしまった場合に検索時間が長くなる．このように，検索者が欲

しい結果件数にあわせた検索範囲の設定が困難だという問題がある．このような問題を解決するためには，検索範囲として数件の特徴量ベクトルが内部に存在するような超矩形領域を指定する手段が必要であり，今後の重要な課題といえる．

6. 結　　び

本論文では，数十次元ないし数百次元の多次元ベクトルを扱う類似画像検索に向けた多次元インデクス手法を提案した．さらに，実装実験を行った結果，590次元という高次元ベクトルを対象にしても，全件走査手法より提案手法の方が高速に類似検索を行えること，検索対象となるデータ数が多くなるほど高速化の効果が高いことを示した．全件走査手法とは，1件ごとに特徴量ベクトルを用いて検索範囲内かどうかを判定するというごく普通の計算方式だが，10次元以上では従来のツリー構造の多次元インデクスよりも高速に類似検索を行えるといわれている．また，類似画像検索において，検索精度が著しく悪化しないことも示した．

謝辞　本研究は，財団法人日本情報処理開発協会（JIPDEC）における「次世代電子図書館システム研究開発事業」の一環として技術開発された内容を含む．

参 考 文 献

1) 片山紀夫，佐藤真一：SR-Tree：高次元データに対する最近接検索のためのインデックス構造の提案，電子情報通信学会論文誌（D-I），Vol.J80-D-I, No.8, pp.703–717 (1997).
2) Berchtold, S., Keim, D.A. and Kriegel, H.-P.: The X-tree An Index Structure for High-Dimensional Data, *Proc. 22nd VLDB*, pp.28–39 (1996).
3) Rudolf, B.: The Universal B-Tree for multi-dimensional Indexing, Techical Report TUM-I9637, Institut fur Informatik, TU Munchen (1996).
4) Weber, R., Schek, H.-J. and Blott, S.: A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces, *Proc. 24th VLDB*, pp.194–205 (1998).
5) 赤間裕樹，小西史和，吉田忠城，谷口展郎ほか：近傍検索向け転置ファイル法における外部キー検索と動的データ追加の実装と評価，情報処理学会論文誌：データベース，Vol.40, No.SIG8 (TOD4), pp.51–62 (1999).

(平成12年 6 月20日受付)
(平成12年10月16日採録)

（担当編集委員　加藤 俊一）

上川　伸彦（正会員）
1972年生．1997年上智大学大学院理工学研究科電気・電子工学専攻修士課程修了．同年，(株）日立製作所入社．マルチメディア情報検索の研究開発に従事．

岩崎　一正（正会員）
1965年生．1992年筑波大学社会工学研究科経営工学専攻博士課程中退．同年，(株）日立製作所入社．マルチメディアデータベースの研究開発に従事．