B1-5

近傍連鎖点列の概念に基づく画像検索の実装と評価

佐藤 康裕† 伊藤 晶規‡
田中 覚§ 遠山 元道‡

†慶應義塾大学大学院 理工学研究科 開放環境科学専攻
‡慶應義塾大学 理工学部 情報工学科 §TIS 株式会社
E-mail: yas@db.ics.keio.ac.jp ito@db.ics.keio.ac.jp
tana@mmm-keio.net toyama@ics.keio.ac.jp

マルチメディアデータベースなどの分野で特徴ベクトル上の類似検索の重要性が増して来ている。それに伴い、多次元データの索引構造の重要性も同様に増している。類似検索では、一般に質問点の近傍点たかだか k 個を求める k-Nearest-Neighbor Query が用いられる。しかしこの手法では、得られた質問結果は質問点からの距離でランキングされただけであるため、ユーザが実際に欲しいデータを見つけるためには、全ての結果を一つ一つ確認する必要が生じる。本研究では、既存の多次元索引構造に対する一次索引を生成する近傍連鎖点列集合を導入する。これにより、検索結果を類似する点毎に分類し、ユーザの求める結果への誘導を容易にする。今回は、近傍連鎖点列集合の概念を用いた画像検索システムを実装し評価・検討について報告する。

キーワード : 近傍連鎖点列、 近傍探索、 類似検索、 多次元空間、 索引技術、 画像検索

Implementation and Evaluation of Image Retrieval Based on Chained Neighborhood Points

Yasuhiro SATO † Akinori ITO ‡
Satoru TANAKA § Motomichi TOYAMA‡

†School of Science of Open and Environmental System, Faculty of Science and Technology, Keio University.
‡Department of Information and Science, Faculty of Science and Technology, Keio University.
§TIS Inc.

The importance of the similarity search on the feature vector has been increasing in fields, such as a multimedia database. In connection with it, the importance of the index structure of multi-dimension data is increasing similarily. In this paper, we introduce Chained Neighborhood Point which generates the primary index to the existing index structure. This classifies a search result for every similar point, and makes it easier to navigate users to their required information. In this time, we implement the system of image retrieval, and evaluate it.

keyword : Chained Neighborhood Point, nearest neighbor search, similarity retrieval, multi-dimensional space, indexing technique, image retrieval

1　はじめに

近年、マルチメディアデータベースなどの必要性の増大に伴い、特徴ベクトル空間上の多次元索引構造の必要性が増して来ている。また、その索引構造に対する効率的な検索手法の必要性も増している。

従来の検索システムの抱える問題点のひとつとして、検索結果の分類が挙げられる。利用者が検索システムに与える質問が多次元空間の 1 点に相当する場合、従来の検索システムではその質問点の近傍点の集合を検索結果として返すケースが多い。しかしこのような場合、検索結果全てが必ずしも利用者の必要とするものではないため、利用者が本当に必要としている検索結果を得るのに手間がかかってしまう。そこで、システム側である程度検索結果を分類して利用者に提示することで、利用者の負担を軽減する必要がある。

本研究では「近傍連鎖点列集合」の概念を採用し、上の条件を満たす画像検索システムを実装する。これについて、近傍連鎖点列集合の静的索引、動的索引双方に対して画像検索を行い、そのパフォーマンスなどの評価を目的とする。

2　類似検索

特徴ベクトル空間上の多次元データの検索には、最近傍問合わせ (Nearest Neighbor Queries)[5] が良く知られている。この最近傍問合わせとは、質問点から最も近い点を見つける問題である。この最近傍探索アルゴリズムを一般化し、質問点から最も近い上位たかだか k 個の点を見つけるように拡張したものが k-Nearest Neighbor 探索アルゴリズムであり、一般に良く用いられている。対象となる点の集合を R-tree のような索引構造木に格納し、その索引構造木に対して検索を行うことで検索結果を得る。

3　近傍連鎖点列集合

近傍連鎖点列 (Chained Neighborhood Points) は、近傍点を順次連結した点列であり、近傍連鎖点列集合 (Chained Neighborhood Points Set) は、近傍連鎖点列の集合である。以下ではその概念について述べる。

3.1　定義

検索対象となるオブジェクト集合を S、S 内のオブジェクトを $o_i \in S$ で表す。各オブジェクトを n 次元空間内の点で表し、S 内の任意の 2 オブジェクト o_i, o_j 間の距離を d_{ij} で表すとき、オブジェクト o_1 を起点とする近傍連鎖点列 (CNP_1) を次のように記述する。

$$CNP_1 = \{o_1 \xrightarrow{d_{1,2}} o_2 \xrightarrow{d_{2,3}} \ldots \longrightarrow o_{d-1} \xrightarrow{d_{(d-1),d}} o_d\} \quad (1)$$

(但し、$o_j = \{o_i \in S | \forall p \in S : d(o_i, o_j) \leq d(o_i, p)\}$)

右向きの矢印 (→) はオブジェクト間の隣接関係を表す。例えば $o_i \xrightarrow{d_{ij}} o_j$ は 2 点間の距離が d_{ij} であり、o_i にとって o_j が最近傍オブジェクトであることを示している。このことから、CNP のオブジェクト間の距離は単調減少するという性質がある。また、o_1 を CNP の起点オブジェクト、o_d を終点オブジェクトと呼ぶ。さらに CNP を構成するオブジェクトの数を CNP の長さとする。距離の記述を省略した場合は CNP を以下のように略記することができる。

$$CNP_1 = \{o_1, o_2, \ldots, o_{d-1}, o_d\} \quad (2)$$

3.2　CNP 基本形

CNP 基本形とは、式 1 で示した CNP の定義そのものである。すなわち、あるオブジェクトを起点オブジェクトとし、その最近傍オブジェクトを順次連結したものを CNP 基本形と呼ぶ。CNP 基本形では、多次元空間上の有限なオブジェクト集合はいくつかの木に分割される (図 1)。CNP 基本形で

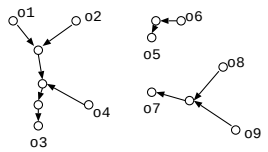

図 1: CNP 基本形

は、終点オブジェクトはいくつかの特定のオブジェクトに集中する。これらの終点オブジェクトを根 (root)、起点オブジェクトを葉 (leaf) と呼ぶ。図 1 では o3,o5,o7 が根にあたり、o1,o2,o4,o6,o8,o9 が葉となる。

3.3 CNP 拡張形

CNP 拡張形とは、CNP の定義を一般化し最近傍点のみの連結ではなく近傍点上位 k 個以内の点を求め、それらの点全てに対し順次近傍点を連結していくものである。

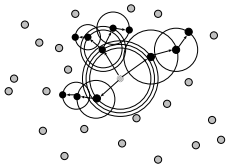

図 2: CNP 拡張形

図 2 は中心のオブジェクトからの CNP 拡張形を生成する場合の例である。

4 検索時索引と静的 CNP 索引

CNP は検索時に生成する方法とあらかじめ静的に生成しておく方法の両方が可能である。

4.1 検索時索引

検索時索引生成の場合、質問点が与えられた時点で CNP の生成を開始する。このため、要求された分岐の数だけ近傍探索を実行しなければならない。一方で、更新作業は索引木のみで済むため、更新のタイミングによる検索結果の不整合やパフォーマンスの低下などは生じないという利点がある。

4.2 静的索引

静的な索引生成の場合、R-tree のような索引木に対して検索時索引と同じ方法で CNP の生成を行い、それを保持しておく。質問点が与えられると、その質問点に対して近傍探索が 1 度だけ行われ、あらかじめ生成しておいた CNP の中で近傍探索で得られた点から始まるものを検索結果として出力する。近傍探索が 1 度で済むため、検索時のパフォーマンスは検索時索引時よりも高い。しかし、データの更新に関しては、索引木そのものの更新だけでなく CNP も更新する必要があるため、頻繁な更新がある場合や頻繁な検索によるアクセスがある場合などはパフォーマンスの低下を招くという欠点が存在する。

4.2.1 静的索引の構築

検索時索引の場合、近傍探索を行い、得られた点に対して再帰的に近傍探索を行う必要があったが、静的索引では再帰的な探索は必要ない。CNP の生成自体は検索時に行い、構築時には索引木内の全ての点に対し近傍点を求めておき、それをデータベースに格納する。データベースには近傍点のオブジェクト ID とともに、その点までの距離も同様に格納しておく。検索時にこのデータベースに対し問合せを行い、CNP を生成する。

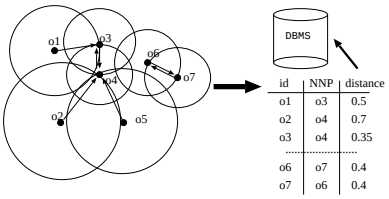

図 3: 静的索引の構築

4.2.2 オブジェクトの挿入

新しいオブジェクトの挿入には索引木と CNP 両方の更新を必要とする。最も簡単な更新方法は全てを作り直すことであるがコストが高くなる。ここでは、オブジェクトの挿入によって影響を受けたオブジェクトを得るために Reverse Nearest Neighbor Query[3] を用いる。Reverse Nearest Neighbor Query(以下 RNN) とは、与えられた質問点に対し、その質問点を最近傍とする点の集合を質問結果とする探索手法である。Nearest Neighbor Query と同様に Reverse k-Nearest Neighbor という拡張も存在する。

挿入されるオブジェクトを q とすると、

1. 索引木に q を挿入する。
2. q の最近傍点を求める。
3. 索引木の q 以外の点に関して、q が最近傍となる点の集合を求める。

4. 3で求まった点の集合に対し、最近傍点を変更する。

という処理を行うことによって、最小限の更新でCNPの再構築が可能となる。

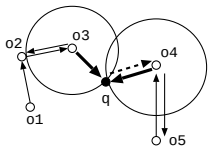

図 4: オブジェクトの挿入

図 4 のように $o1$〜$o5$ のオブジェクトがあるところに点 q が挿入されたとすると、まず q が挿入される。次に q の最近傍点を求めるとここでは $o4$ となり、図 4 の破線の矢印のように CNP に格納される。$o1$〜$o5$ に対し RNN Query を実行すると、$o3, o4$ が q の RNN であることが分かる。そこで、もともとそれぞれ $o2, o5$ であった $o3, o4$ の最近傍点を q に変更し、CNP を更新する。

4.2.3 オブジェクトの削除

削除も挿入と同様に RNN Query に用いて影響を受けるオブジェクトを検索し、更新する。

削除されるオブジェクトを q とすると、

1. q からの CNP を削除する。
2. 索引木から q を削除する。
3. 残された点の中で、q が最近傍であった点の集合を求める。
4. 3で得られた点の集合全てに対し新たに最近傍点を求め、CNP をその最近傍点で更新する。

という処理でできることになる。

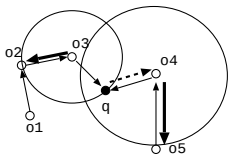

図 5: オブジェクトの削除

挿入の時と同様に、図 5 のような点の配置で点 q を削除する場合、まず q から出ている破線の矢印にあたる CNP を削除する。次に q を索引木から削除する。ここで、挿入時は RNN Query によって q を最近傍とする点を求めたが、削除時は CNP 側で q を参照していた点の集合を直接求めることができる。図 5 の例では $o3, o4$ が q を参照していたので、$o3, o4$ が求まることになる。最後に得られた $o3, o4$ に対し最近傍点を求める。それぞれ、$o2, o5$ となり太矢印にあたる CNP が新たに格納される。

4.2.4 CNP 拡張形の場合の静的索引

4.2.3 節までで示した静的索引の構築・更新アルゴリズムは CNP 基本形のみ、つまり最近傍点のみを考慮にいれたアルゴリズムだった。しかし、アルゴリズムのどの部分も最近傍点の部分を近傍点上位たかだか k 個と変更することで CNP 拡張形にも同様に適用可能である。RNN Query に関しても、先に述べたように Reverse k-Nearest Neighbor Query が存在するので問題無く適用できる。

静的索引では静的であるがゆえに、あらかじめ生成された索引の範囲を越えて検索することはできない。問題となるのは、索引構築時と検索時の k の値をそれぞれ索引の次数、検索の次数と呼ぶことにすると、検索の次数が索引の次数を上回るときには検索できないということである。これに関しては、現在は検索時に制限をかけることで対応するのがもっとも簡単な対処方であるが、索引中のオブジェクトまでの円の外側の空間に対し近傍探索を実行し、(検索の次数)-(索引の次数) の数だけ新たにオブジェクトを得ることで解決できる。

5 CNP の実装と評価

CNP は特定の索引木に依存するものではなく、どんな索引木に対しても実装が可能である。今回は SR-tree[4] を索引木に採用し、実装を行った。また、実験は以下の環境で行った。

CPU	celeron 600MHz
メモリ	128MByte
OS	Linux-2.4.17(kernel version)

5.1 静的索引の生成時間

まず、静的索引の生成時間の測定を行った。静的索引の生成には探索対象とする近傍点の数 k を決定する必要があるが、今回 k は 10 とした。

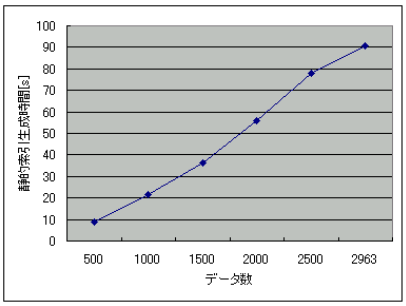

図 6: 静的索引の生成時間 $(k = 10)$

図 6 に静的索引の生成時間 $(k = 10)$ の測定結果を示す。結果から、データ数と生成時間の間にはほぼ比例の関係があることが分かる。しかし、3000 件のデータに対して 90 秒もかかっており、絶対的な処理時間としてはデータが増加した場合を考えるとあまり良い結果だとは言えない。また、この結果からもデータ更新時には差分更新が必須であると言うことができる。

5.2 検索時間の比較

次に、静的索引と検索時索引双方で検索時間の測定を行い、処理時間の比較を行った。質問点から探索する近傍点の数 k は 5、探索の結果見つかった点から更に探索する点の数 s を 3 として測定を行った。

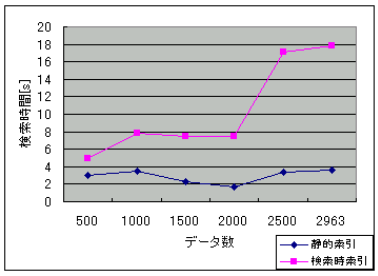

図 7: 検索時間の比較 $(k = 5, s = 3)$

図 7 に検索時間の測定結果を示す。◆が静的索引時、■が検索時索引時の検索時間である。図 7 から明らかな通り、静的索引ではデータが増加しても検索時間がほぼ一定である。一方、検索時索引ではデータ数 2000 件まではほぼ一定だが、それを超えると処理時間が急激に増加している。考えられる原因としては、検索時索引では検索中の点を全てメモリ上に蓄えているが、データ数の増加に伴うメモリ不足が挙げられる。

5.3 k の値による検索時間の変化

質問点からの探索点の数 k を変化させた場合の検索時間の変化について測定した。図 8、図 9 に静的索引、検索時索引双方で、k の値を 5 から 7 に変化させた時の測定結果を示す。

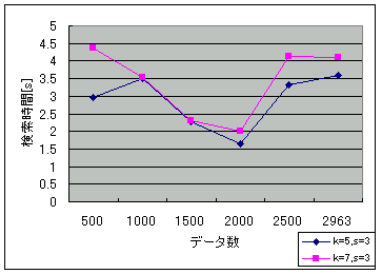

図 8: 静的索引時の変化

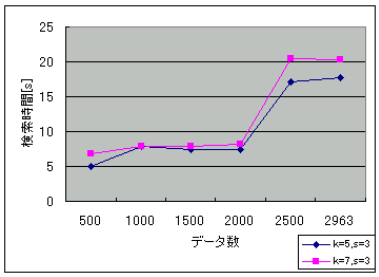

図 9: 検索時索引時の変化

図 8、図 9 で、◆が $k = 5$ の場合、■が $k = 7$ の場合の測定結果である。この結果ではどちらの検索方法でも k の値の変化前と変化後でほぼ同じ軌跡をたどっており、k の値の変化は検索結果に影響をほとんど与えないということが分かる。しかし、データ数が 2500 件をこえたところから多少ではあ

るが $k = 7$ の場合の方が検索に時間がかかっており、データ数が更に増えた場合はもっと顕著にこの傾向が出ることも考えられる。

また、k を固定し s の値を増やした場合の実験も行ったが、$k = 5$ で s を 3 から 5 に増やした場合、前の実験では多くても数秒で検索を終了していた静的索引でも、500 件のデータに対し 519 秒もかかるという結果となった。このことから、s の値の増加は検索時間に多大な影響を与えることが分かり、その決定は慎重にしなければならないと言える。s の変化が与える影響が大きいのは、根ではなく枝の増える割合だからであるが、これは CNP があまり長くないときにはそれほど影響は大きくない。検索結果を利用するユーザにとって、検索結果をあまりに多くたどらなければたどり着けない検索結果は必要ないと考えれば、CNP の長さの最大値をあらかじめ設定しておくことで、s が大きい場合でも影響が大きくなる前に検索が打ち切られることになり、問題を解決できる。

6 画像検索システム

画像検索を検索するにあたり、内容検索ではなく特徴量検索であることから、特徴量の抽出は重要な問題である。画像の特徴量抽出の方針としては、画像を分割しそれぞれの領域の色情報とその領域の周波数を抽出している。また、デジタルカメラの EXIF 情報を利用した画像の検索および画像整理支援も実装した。

6.1 検索に用いる画像の特徴量

画像の内容に基づく検索を行う場合、特徴量として利用可能なものは色、模様、構図等が挙げられる。

6.1.1 色

画像の特徴量として、図 10 のように画像を分割し、画像全体及び画像の一部の領域からそれぞれ 8 色のカラーヒストグラムを作成した。

図 10 の画像分割の目的はそれぞれ、

(1) 画像全体の色合いなどに基づき類似検索を行うことで、海や山といった背景画や、カーテンなどのパターン画像に適した全体画検索を行う。

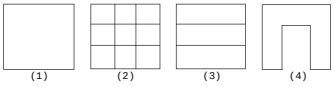

図 10: 画像領域の分割パターン

(2) 画像中のさまざまな場所に存在する比較的小さな被写体に注目し、その被写体 (画像オブジェクト) に基づいた類似検索を行う。

(3) 空や海などといった上下に大きな領域を持つと考えられる比較的大きなオブジェクトを抽出し、それに基づく類似検索を行う。

(4) 中央部分をマスクすることにより、背景情報に注目した類似検索を実行する。

6.1.2 主要な直線

風景画像における地平線、水平線や建築物などによる直線が画像の構図に強い影響を与えることから、画像内の主要な直線の有無を考慮する。画像から発見される特徴的な直線の中で、その直線の長さが画像の長辺の 3 分の 2 を超えるものが存在する場合としない場合を特徴量としている。

6.1.3 Exif 情報

画像の表層的な特徴以外に、デジタルカメラによって撮影された画像には Exif 情報が添付されている。この情報を用いて検索を行うことで、意味的に類似した画像の検索が実行できる。本研究では以下にあげる Exif データを用いて検索を行った。

- 縦幅、横幅
- 焦点距離
- 露光時間
- 撮影日時
- Flash の有無
- F 値

各 Exif データは、検索対象となる画像オブジェクト群の持つデータ中の最大値、最小値を用いて [0,1] の値として正規化している。

6.2 実行例・評価

画像検索システムの検索実行例を図 11 に示す。

検索の実行にはキーとなる画像名に加え、質問点から探索する点の数 k、また、発見されたオブジェクトから探索する点の数 s について入力を行う。

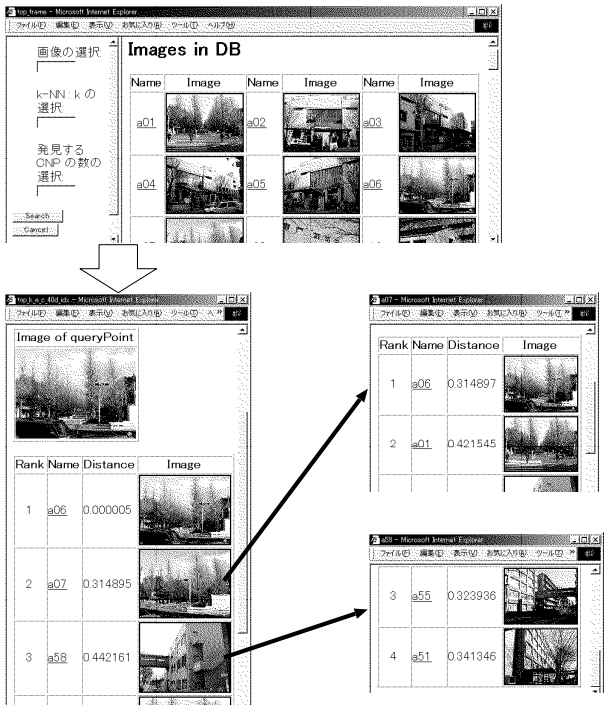

図 11: 画像検索の実行画面

図 11 のように画像を選択することで、選択された画像に類似した画像が検索される。これにより検索結果を類似するもの同士で分類する事が可能になり、検索効率の向上に役立てることが出来る。

7 結論

本研究では、近傍連鎖点列 (CNP) の静的索引、検索時索引の両方を実装し、評価した。測定結果では、検索時索引の方が静的索引に比べ数倍速いという結果が出たが、静的索引に関してはまだ改善するべき点が多いことが分かった。また、近傍連鎖点列を利用した画像検索システムを実装し、その実行例を示した。

今後の課題としては、静的索引の索引方法の改善による処理時間の短縮。利用するアプリケーションに依らない出力形式の検討などが挙げられる。

参考文献

[1] 田中 覚, 遠山 元道: 多次元空間における近傍連鎖点列を用いた類似検索システム, Database Engineering Workshop 2001,4B-3

[2] 佐藤 康裕, 田中 覚, 遠山 元道: 近傍連鎖点列探索における静的索引, Database Workshop 2001,...

[3] Flip Korn, S. Muthukrishnan: Influence Sets Based on Reverse Nearest Neighbor Queries, Proc. ACM SIGMOD Int. Conf. on Management of Data, 2000, pp.201-212.

[4] Norio Katayama, Shin'ichi Satoh.: The SR-tree: An Index Structure for High-Dimensional Nearest Neighbor Queries, Proc. ACM SIGMOD Int. Conf. on Management of Data, 1997, pp.369-380.

[5] Nick Roussopoulos, Stephen Kelley, Fedeic Vincent: Nearest Neighbor Queries, Proc. ACM SIGMOD Int. Conf. on Management of Data, 1995, pp.71-79.

[6] 串間 和彦, 赤間 浩樹, 紺谷 精一, 木本 晴夫, 山室 雅史: オブジェクトに基づく高速画像検索システム : Exsight, 情報処理学会論文誌, 1999, vol.40, No.2

[7] A. Guttman: R-trees: A Dynamic Index Structure for Spatial Searching, Proc. ACM SIGMOD Int. Conf. on Management of Data, 1984, pp.47-57.