

木構造型インデックスを用いた 近似 k 最近傍グラフによる近傍検索

岩崎 雅二郎^{†1}

大量の多種多様なデータを高速に検索するには空間インデックスが不可欠となる。空間インデックスの 1 つである k 最近傍グラフによるインデックス (kNNG) では最近傍検索の高速性が確認されているが、インデックス生成のコストがきわめて高いという問題がある。インデックス生成時に生成途中のインデックスを用いて k 最近傍検索を行うことで kNNG を近似するインデックスを高速に生成する ANNG (Approximate k-Nearest Neighbor Graph) を筆者はすでに提案している。しかし、ANNG ではランダムに選択したノードからグラフを探索するのでグラフが大量のノードを有する場合に探索コストが増加するという問題点がある。そこで、本稿では ANNG において木構造型のインデックスを利用することにより低コストで近傍ノードを探索できる方法を提案し、一様分布データおよび画像特徴量を用いて提案手法が ANNG よりコストを低減できることを確認した。

Proximity Search using Approximate K Nearest Neighbor Graph with a Tree Structured Index

MASAJIRO IWASAKI^{†1}

Spatial indexing technology is indispensable for efficient retrieval of large amounts of various data. K-nearest neighbor graph (kNNG), which is one of the graph-based index structures, is known to significantly reduce the search time, however, the cost of index construction is very high. In our previous work, we proposed Approximate k-Nearest Neighbor Graph (ANNG) which uses the partially constructed index to perform the k-nearest neighbor search during index construction. However, when ANNG has large number of nodes, the cost of searching neighbor nodes from the node selected at random is high. In this paper we address this issue, and propose a method that uses tree structured index to reduce the cost of searching neighbor nodes. We experimentally show that the proposed method outperforms ANNG on both the uniformly distributed data and the actual image feature data.

1. はじめに

インターネットの普及につれて画像、動画、音声などの多様なデータを日常的に利用するようになった現在、大量に存在するデータを高速に検索する技術が注目されてきている。条件に適合するデータを検索するには対象となる全データを評価すればよいが、データ量の増加とともに検索時間が増加し大規模なデータに対しては現実的な時間で処理ができなくなる。そこで、大規模なデータには検索用のインデックスが不可欠となる。画像検索を例にすると画像から抽出した多次元の特徴量を高速に検索するには、特徴量空間を検索する空間インデックスが必要になる。

空間インデックスには大きく分けて多次元空間インデックスと距離空間インデックスの 2 種類がある。多次元空間インデックスは次元の要素を意識したインデックスであり、空間で定義された距離関数に対してインデックスのアルゴリズムが定義される。したがって、一般に距離関数が変わるとインデックスのアルゴリズム自体を変えなくてはならない。一方、距離空間インデックスは次元の要素を意識せず、どのような距離関数でも適用できるので応用範囲が広い。また、距離空間インデックスの検索処理の時間は主に、記憶装置からデータをロードする時間、および、距離空間内での距離計算の時間、からなる。近年、メモリが安価で大規模になり大量のデータを高速なメモリに配置することが可能になった反面、画像検索のように超多次元で、かつ、空間上での距離計算のコストの高い応用が増加している。したがって、記憶装置からデータをロードする時間よりも、距離計算の回数を削減することが課題となってきている。また、扱うデータ量が総じて大規模になってきており、その結果、インデックスの生成コストを削減することも課題である。

以上のことから、検索時およびインデックス生成時の距離計算回数を削減可能な距離空間インデックスである ANNG¹⁾ を筆者は提案した。ANNG はグラフ構造型によるインデックスであり、多次元データに対しても生成コストを抑えつつ距離計算回数を削減することが可能である。ANNG ではグラフをたどり検索条件を満たすノードを探索するが、ランダムに選択したノードからグラフの探索を開始するのでノードが増えるほど検索時のコストが増大する傾向がある。そこで、本稿では探索を開始するノードを効率良く選択できるように木構造型インデックスを利用することで、ANNG においてノードが増加したときのコストの

^{†1} Yahoo! JAPAN 研究所
Yahoo! JAPAN Research

増大を抑制する方法を提案し評価する。

2. 関連研究

空間インデックスは次元数が少ない場合には効果的に機能するが、次元数が増えれば増えるほどインデックスの効果がなくなる「次元の呪い」という現象が生じる。この問題を克服すべく様々なインデックスが提案されている。

多次元空間インデックスでは、木構造型の R-tree²⁾, R*-tree³⁾, kd-tree⁴⁾, quadtree⁵⁾, SS-tree⁶⁾, SS+-tree⁷⁾, X-tree⁸⁾ や次元ごとに圧縮して線形探索を高速化する VA-File⁹⁾ が提案されている。また、検索結果に漏れが生じることを許容して高速に検索する近似検索の研究があり、kd-tree に近似検索を適用した ANN¹⁰⁾ や、ハッシュを用いた LSH¹¹⁾ といった近似検索のインデックスが提案されている。

これらの多次元空間インデックスは様々なデータに対して応用されているが、画像検索に応用する場合には、画像から抽出される特徴量の多様性に起因する問題が生じる。検索精度を重視すると、カラーヒストグラムの特徴量の距離には 2 次形式 (quadratic form) 距離¹²⁾ を、色の特徴量には色差式に基づく距離を、そしてテキストの特徴量には L_1 距離 (市街地距離) を利用する、といったように様々な距離の組合せが必要になる。多次元空間インデックスは一般に L_p 距離を対象としているので、このような多様な特徴量に対して単純には多次元空間インデックスを適用できない。そこで、平均色による多次元空間インデックスによって検索した後に誤検索を除去する方法¹³⁾ や 2 段階の処理により 2 次形式距離に対応したインデックスを生成する方法¹⁴⁾ などがある。

画像から抽出した特徴量空間の多様な距離関数に対して多次元インデックスは容易には適用できないが、距離空間インデックスはどのような距離関数にもインデックスのアルゴリズムを変更せずにそのまま適用できる。しかし、多次元空間インデックスと同様に距離空間インデックスにも「次元の呪い」が存在する。距離空間インデックスの場合には次元を考慮しないが、一般に次元数が多くなればなるほど近接する点の個数が減少する、つまり、空間上の点が距離的に疎に分布することになり、距離空間インデックスも多次元空間インデックスと同様に次元数が多くなると高速に検索することが困難になる。

多次元空間インデックス同様、距離空間インデックスの研究も継続的に行われ、様々なインデックスが提案されている。多次元空間インデックスと同様に木構造型の GH-tree¹⁵⁾, GNAT¹⁶⁾, vp-tree¹⁷⁾, mvp-tree¹⁸⁾, M-tree¹⁹⁾ といった方法が提案されている。筆者は vp-tree を改良し追加更新が可能な動的なインデックスである dvp-tree²⁰⁾ を提案した。ま

た、あらかじめ各点間の距離を計算し、その距離データを用いて距離計算せずに対象となる点の絞り込みを行う AESA²¹⁾, LAESA²²⁾ といった方法が提案されている。

その一方、検索時における距離計算回数の削減においてグラフ構造型のインデックスの有効性が確認されている。グラフ構造型のインデックスの場合には、エッジをたどりながら条件に適合するノードを探索して検索結果を得る。各ノードに付与されているエッジ数が多い場合にはグラフの生成コストが大きくなり、かつ、探索コストも増えるが、検索精度は高くなる。逆にエッジが少ない場合にはグラフの生成コストが減り、かつ、探索コストも減るが、検索精度は低くなる傾向がある。ポロノイ図において隣接するノードをエッジで結合したグラフであるドロネーグラフ²³⁾ ではエッジ数が比較的少なく、探索コストを低減でき、かつ、高い検索精度を実現できる。

ユークリッド距離の場合にはドロネーグラフの生成が可能なので検索用のインデックスとして利用できるが、多次元データではドロネーグラフの生成コストが高くなる。ユークリッド距離ではない場合にはドロネーグラフを生成することが困難とされている。しかも、次元数が増えるほどエッジ数は増加するので、ドロネーグラフが生成できたとしても、そのエッジ数が膨大となるので実用的とはいえない。

そこで、ドロネーグラフではないが、その一部を構成する木構造型のインデックスである sa-tree²⁴⁾ や、擬似的にドロネーグラフを生成する最近傍検索の方法²⁵⁾ が提案されている。また、k 最近傍グラフにより最近傍のノードを検索する kNNG²⁶⁾ も提案されている。kNNG は各ノードから k 個の最近傍のノードへの有向エッジで構成されたグラフである。kNNG は簡便なグラフ構造でありながら、最近傍検索において少ない距離計算回数で、かつ、高い検索精度を実現している。その一方で kNNG の生成コストはきわめて高い。kNNG 生成時の距離計算回数を抑制する方法^{27),28)} が提案されているが、いずれもユークリッド空間を対象とした方法である。距離空間を対象とした方法²⁹⁾ もあるが、距離計算回数を削減するのではなく並列処理により処理時間を短縮することに主眼をおいた方法である。

このようなことから、筆者は kNNG のグラフ構造を近似する ANNG¹⁾ を提案した。ANNG は距離空間上で k 最近傍グラフを近似するグラフであり、各ノードは無向エッジにより近傍ノードと結合している。k 最近傍グラフの生成時に近傍ノードを探索しなければならないが、近傍ノードの探索コストが高いので、k 最近傍グラフの生成コストはきわめて高くなる。しかし、ANNG ではインデックスの生成時の探索に生成途中のインデックスを用いることで近傍ノードを高速に検索するので、インデックスの生成コストを低減できる。その結果、ANNG は kNNG とほぼ同等またはそれ以上の検索精度にもかかわらずインデックス生成

のコストを低減させている。

ANNG では検索時に 2 段階の処理を行う。1 段目ではグラフからランダムにノードを選択し、そのノードを起点としてグラフをたどり検索点に対して単一の近傍ノードを探索する。2 段目ではその近傍ノードから網羅的にグラフをたどりつつ、検索条件を満足するノードを探索し検索結果を得る。1 段目の処理ではノードからランダムに選択したノードを起点としているので探索コストは登録数の増加に比例して増大する。また、ランダムに選択した起点となるノードから検索点までの距離に比例して探索コストが増大するので、各検索ごとにコストが変動する。これらの問題を解消するために、本稿では 1 段目の処理に木構造型のインデックスを用いる方法を提案し評価する。

3. ANNG インデックス

ユークリッド空間においてグラフが以下の条件 1 を満たすならば、エッジで結合された近傍ノードを探索すること（局所探索）によって最近傍検索が可能である²⁴⁾とされている。

条件 1 G をグラフ、 p を G のノード、 $N(G, p)$ を G における p の近傍ノードの集合、 q を検索点、 $d(p, q)$ を p, q 間の距離とすると、ノード p が $N(G, p)$ に属する任意のノード x に対して $d(p, q) < d(x, q)$ が成り立つならば、 G に属する任意のノード x に対して $d(p, q) < d(x, q)$ が成り立つ。

条件 1 を満足する場合、Algorithm 1 に示す最良優先探索により最近傍検索が可能である。Algorithm 1 の G をグラフ、 q を検索点、 s を探索起点ノード（グラフの探索を開始する起点となるノード）とする。各ノードに全ノードへのエッジが付与されている完全グラフも条件 1 を満たすが、あるノードの近傍ノードは全ノードとなるので、全件に対して線形探索することと同等となる。ドロネーグラフも条件 1 を満足する²⁴⁾とされており、かつ、比較的少ないエッジで構成される。ANNG の場合には条件 1 を満足しない近似 k 最近傍グラフなので、Algorithm 1 では最近傍のノード以外のノードが返る場合が存在する。そこで、異なる探索起点ノードにより何回か最近傍検索を繰り返し、最も近いノードを最近傍のノードとする必要がある。

ANNG の範囲検索および k 最近傍検索では 1 段目の処理として、この最近傍検索により検索点の近傍ノードを探索する。なお、最近傍検索時の探索起点ノードの決定方法は様々考えられるが、ANNG では全ノードからランダムに選択する方法を採用している。

ANNG の範囲検索の場合には、まず 1 段目の処理として前述の最近傍検索 (Algorithm 1) を行い 2 段目の探索起点ノードとなる範囲内の単一の近傍ノードを見つけ、次にその探索

Algorithm 1 NearestNeighborSearch(G, q, s)

```

 $p \leftarrow s$ 
 $P \leftarrow N(G, p)$ 
 $c \leftarrow \operatorname{argmin}_{x \in P} d(x, q)$ 
if  $d(c, q) < d(p, q)$  then
     $p \leftarrow \text{NearestNeighborSearch}(G, q, c)$ 
end if
return  $p$ 

```

起点ノードからエッジをたどって網羅的に検索範囲内のすべてのノードを探索する。条件 1 を満足する場合にはこの方法で十分であるが、ANNG の場合には条件 1 を満足しないので、最近傍検索において範囲内の近傍ノードが見つからない場合がある。そこで、ランダムに選択したノードから最近傍検索を行い範囲内の近傍ノードが見つかるまで繰り返す。また、2 段目の処理として範囲内のノードの網羅的な探索においても、最近傍のノードへの経路が 1 度範囲外に出た後に再度範囲内に戻る場合を想定しなければならない。そこで、検索範囲より広い範囲を探索するために探索する範囲（超球）の探索半径 r_s を式 (1) のように決定する。

$$r_s = (1 + \epsilon)r \quad (1)$$

ただし、 r は検索範囲を示す探索半径、 ϵ は探索半径係数とする。この探索半径内のグラフを網羅的に探索することで検索結果のノードを収集する。この探索半径 r_s を用いた範囲検索の 2 段目のアルゴリズムを Algorithm 2 に示す。 r を探索半径、 R を検索結果集合とし、 C は同じノードを何度も探索しないように判定するための集合である。 R, C の初期値には空集合を設定する。

ANNG の k 最近傍検索では、一般的な k 最近傍検索の方法を採用している。つまり、探索半径の初期値を無限大として、探索途中における検索結果の最遠のノードの半径により探索半径を狭めつつ前述の範囲検索を行う。なお、範囲検索と同様に 1 段目の処理として最近傍検索を行うが、範囲検索とは異なり探索半径の初期値が無限大であることから、最近傍検索により必ず範囲内（無限大）に入るので最近傍検索は 1 回で十分となる。k 最近傍検索のアルゴリズムを Algorithm 3 に示す。ただし、 k_s は検索数、Random(G) は G からランダムに 1 ノードを返す関数である。Algorithm 3 から呼び出される KnnSearch 関数 (k 最

Algorithm 2 RangeSearch(G, q, s, r, R, C)

```

 $r_s \leftarrow (1 + \epsilon)r$ 
for all  $p \in N(G, s)$  do
  if  $p \notin C$  then
     $C \leftarrow C \cup \{p\}$ 
    if  $d(p, q) \leq r$  then
       $R \leftarrow R \cup \{p\}$ 
    end if
    if  $d(p, q) \leq r_s$  then
      RangeSearch( $G, q, p, r, R, C$ )
    end if
  end if
end for

```

Algorithm 3 AnngKnnSearch(G, q, k_s, R)

```

 $s \leftarrow \text{Random}(G)$ 
 $s \leftarrow \text{NearestNeighborSearch}(G, q, s)$ 
 $r \leftarrow \infty$ 
 $R \leftarrow \emptyset$ 
 $C \leftarrow \emptyset$ 
KnnSearch( $G, q, s, r, k_s, R, C$ )

```

近傍検索の 2 段目の処理) を Algorithm 4 に示す。

ANNG では、このようにして最近傍検索、範囲検索、k 最近傍検索が可能であるが、条件 1 を満たさないで、いずれも近似検索となり検索漏れが発生する可能性がある。

ANNG のインデックスを生成するには、既存のグラフに 1 ノードずつ無向エッジにより逐一連結する。こうすることで連結グラフを保証することができる。ここで、1 ノードを連結するために k 個の最近傍のノードを求める必要があるが、全ノードとの距離を計算したう

Algorithm 4 KnnSearch(G, q, s, r, k_s, R, C)

```

for all  $p \in N(G, s)$  do
  if  $p \notin C$  then
     $C \leftarrow C \cup \{p\}$ 
    if  $d(p, q) \leq r$  then
       $R \leftarrow R \cup \{p\}$ 
      if  $|R| > k_s$  then
         $R \leftarrow R - \{\text{argmax}_{x \in R} d(x, q)\}$ 
      end if
      if  $|R| = k_s$  then
         $r \leftarrow \max_{x \in R} d(x, q)$ 
      end if
    end if
     $r_s \leftarrow (1 + \epsilon)r$ 
    if  $d(p, q) \leq r_s$  then
      KnnSearch( $G, q, p, r, k_s, R, C$ )
    end if
  end if
end for

```

えで k 個の最近傍のノードを求める処理はきわめてコストの高い処理である。そこで、生成過程にあるインデックスを利用して前述の k 最近傍検索 (Algorithm 3) を用いて検索することにより低コストで k 個の最近傍のノードを求めることができる。ANNG 生成の手順を Algorithm 5 に示す。ただし、 P を登録点の集合、 k は各ノードからエッジで結合される近傍のノードの数である。

生成時のエッジ数を 3 とした場合の kNNG の例を図 1 に ANNG の例を図 2 に示す。図中の数字は追加順序を示す。図 1 のように kNNG では非連結グラフが生成されるのに対して ANNG では連結グラフが生成される。ANNG は無向エッジを逐一追加するので、たとえば 2 つのノードのみが追加された時点では、第 1 のノードには 1 本のエッジしか存在し

Algorithm 5 AnngCreate(G, P, k)

```

for all  $q \in P$  do
  if  $G = \emptyset$  then
     $G \leftarrow \{q\}$ 
  else
    AnngKnnSearch( $G, q, k, R$ )
     $G \leftarrow G \cup \{q\}$ 
     $N(G, q) \leftarrow R$ 
    for all  $p \in R$  do
       $N(G, p) \leftarrow N(G, p) \cup \{q\}$ 
    end for
  end if
end for

```

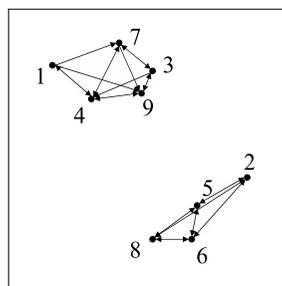


図 1 kNNG
Fig. 1 kNNG.

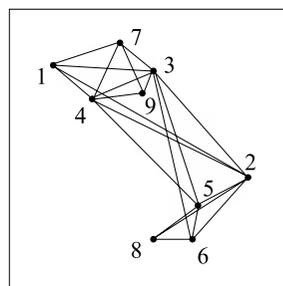


図 2 ANNG
Fig. 2 ANNG.

ないが、順次ノードを追加していくうえで、第 1 のノードの近傍にノードが追加されると無向エッジが第 1 のノードにも追加されることになり、結果的に各ノードには近傍のノードへのエッジが付与されることになる。エッジ数を k 本とした場合でも ANNG では実際には 1 つのノードから k 本以上のエッジが生成されることになり、各ノードのエッジ数は k とはならない。したがって、 k 本のエッジを有する kNNG とはエッジ数では異なるが、kNNG と同様に各ノードから近傍のエッジが生成されることになる。なお、初期に追加したノード

にはエッジが多く付与されることになるが、全体のエッジ数は最大 kn に抑えられる。このように生成された ANNG は kNNG とは同一のグラフ構造にはならないが、検索時の漏れを許容するインデックスとして十分な検索精度が実現できる。

4. 木構造型インデックスによる近傍ノードの探索

ANNG では 1 段目の処理としてランダムに選択したノードを起点にグラフ上で最良優先探索により検索点に対する単一の近傍ノードの探索を行う。この方法では登録数が少ない場合には問題とならないが、登録数に比例して起点から近傍ノードへの経路が長くなり、結果として探索コストが増加する。2 段目の検索が局所的な探索なので登録ノード数にあまり依存しないことから、登録数が増大すると 1 段目のコストが無視できない状況となる。また、ランダムに選択したノードが偶然にも検索点から経路として遠方である場合には探索コストが増大するので、検索ごとにコストが極端に増減する傾向がある。また、ANNG では前述のようにインデックス生成時に検索処理を行う。したがって、検索コストの低減はインデックス生成コストの低減にも寄与するので、検索コストの改善は重要である。

1 段目の処理は最近傍検索にほかならないので、何らかのインデックスを用いることにより効率良く検索できると考えられる。ただし、そのインデックスは以下の条件を満足しなければならない。

- (1) ANNG は距離空間のデータを対象とするので、距離空間インデックスでなければならない。
- (2) ANNG が次元の呪いに耐性があるので、次元の呪いに耐性がなければならない。
- (3) ANNG にデータを登録するときには 1 データごとに ANNG に登録しては検索できない必要がある。つまり、追加登録が可能な動的なインデックスでなければならない。

以上の条件のうち、次元の呪いへの耐性のあるインデックスは、本稿の最大の目的とするところであり、このようなインデックスはないといってもよい。しかし、近似最近傍点であれば次元の呪いの影響を受けにくいインデックスが存在する。

たとえば木構造型インデックスでは次元の呪いは探索枝の削減ができなくなることに原因がある。木構造型インデックスでは各ノードに対応する部分空間が重複するインデックスと重複しないインデックスが存在する。重複する場合には、ルートノード（木構造の起点となるノード）からリーフノード（下位ノードが存在しない末端のノード）へたどる場合には重複するすべてのインターナルノード（下位ノードが存在するノード）を探索しなければならず、次元の呪いの影響を受けやすくなる。しかし、重複しない場合にはルートノードから

つねに各インターナルノードから 1 つの枝をたどればよく、次元の呪いの影響を受けない。ただし、到達したリーフノードの部分空間は必ず検索点を包含するが、かといって必ずしも最近傍点を含むわけではない。しかしながら、1 段目の最近傍探索の目的は 2 段目の起点となるノードを探索するためなので、必ずしも最近傍が得られる必要はない。

したがって、第 2 の条件である、次元の呪いへの耐性は、ノード空間の重複がない木構造型インデックス、とよい換えることができる。本稿では、これらの条件を満足するインデックスとして筆者が以前提案した dvp-tree を用いることとした。なお、dvp-tree を用いた ANNG を既出の ANNG と区別するために以降 ANNGT と呼ぶこととする。

動的に点を vp-tree に追加できるように改良を加えた方法が dvp-tree である。vp-tree はあらかじめすべての点が与えられることを前提とし、vantage point と呼ばれる点を中心とした超球により空間を分割することで木構造を生成する。まず、超球の内側と外側の部分空間にすべての点がそれぞれ同数に分配されるように空間を分割する。さらにその各部分空間を新たな超球により同様に分割し点を分配する。この処理を再帰的に行うことで木構造の vp-tree を生成する。単一の vantage point に対して 2 つ以上の異なる半径により 3 つ以上の部分空間に分割することも可能である。各部分空間は vp-tree のノードに対応し、リーフノードにのみ点が属することになる。一方、dvp-tree では初期時には全空間に対応する単一のリーフノードしかない。点を追加するときに、まず木構造をたどり対応するリーフノードを求めて、そのリーフノードに点を追加する。リーフノードが属する点の数が最大数（リーフノードに属することができる点の最大個数）を超えた場合には、そのリーフノードの空間を vp-tree と同様に分割してノードを生成することで木構造が成長する。生成方法は異なるが木構造の構成は vp-tree と dvp-tree は同様である。

dvp-tree のインターナルノードの分岐数を 2、リーフノードに関連付けられるグラフのノードの最大数を 3 とした ANNGT のデータ構造を図 3 に示す。グラフ構造の ANNG の各ノードが dvp-tree のリーフノードに関連付けられている。図 3 における dvp-tree によるリーフノードの部分空間とグラフの関係を図 4 に示す。円弧で分割されている領域がリーフノードの部分空間を示し、直線がグラフを示している。なお、円弧上は内側の空間に属するとしている。各リーフノードは部分空間に対応付けられており、その部分空間に属するノードがリーフノードに関連付けられている。したがって、検索時にはルートノードより検索点を包含するノードを順次たどることでリーフノードに達し、リーフノードに関連付けられているノード集合が検索点に対する近傍ノード集合となる。そのノード集合を用いて 2 段目の処理を行う。ANNG では単一の近傍ノードから 2 段目の探索を行うが、ANNGT で

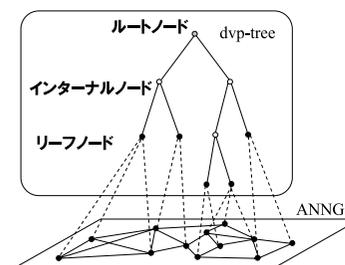


図 3 ANNGT データ構造
Fig. 3 Data structure of ANNGT.

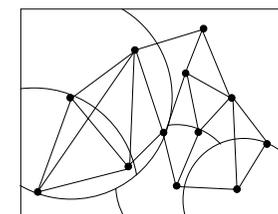


図 4 dvp-tree による部分空間
Fig. 4 Subspaces of dvp-tree.

は複数のノードを起点として 2 段目の探索を行うこととする。範囲検索において探索範囲が小さい場合には、リーフノードにより得られたノードがいずれも範囲内に属さない場合もありうる。このような場合には、範囲外となったノードから ANNG の 1 段目の処理と同様に最良優先探索により範囲内の近傍点を探索する。

ANNGT のインデックスの生成では、前述の検索により登録点から近傍のノードを探索し、その近傍ノードへのエッジおよび登録点を示すノード（登録ノード）をグラフに登録する。この処理は ANNG の登録処理と同一であり、Algorithm 5 の AnngKnnSearch に代わって ANNGT の最近傍検索を呼び出すことになる。ただし、さらに登録点を包含するリーフノードに登録点を追加する操作が必要となる。追加すべきリーフノードは登録時における dvp-tree による検索ですでに特定されているのでそのリーフノードに登録点を追加する。リーフノードへの追加の処理は前述の dvp-tree の登録処理と同一である。

5. 評価実験

一様分布データを用いたインデックスの生成コストおよび検索精度、そして、実際の画像特徴量を用いた検索精度に関して kNNG, ANNG, ANNGT の比較評価を行った。また、一様分布データを用いて既存の代表的な距離空間インデックスの検索精度との比較評価も行った。

評価に使用した一様分布データと画像特徴量のデータ数はいずれも 10 万件である。一様分布データは範囲 $[0.0, 1.0]$ の浮動小数点の要素からなる 20 次元のベクトルデータであり、 L_2 距離を距離関数として用いた。画像特徴量は、画像データから抽出したカラーヒストグラム、色およびエッジの分布、テクスチャで構成される特徴量である。特徴量の総次元数は

1,228 であり、各次元は 1 バイトの整数型である。式 (2) のように各特徴量の距離（色差や L_1 距離など）の加重平均を距離関数として用いた。

$$d(x, y) = \frac{1}{n} \sum_i^n w_i d_i(x_i, y_i) \quad (2)$$

なお、 x_i, y_i は i 番目の単一特徴量、 x, y はそれぞれ x_i, y_i で構成される複合特徴量、 $d_i(x_i, y_i)$ は i 番目の単一特徴量の距離関数、 w_i は i 番目の単一特徴量の重み付け、 n は単一特徴量の総数である。

dvp-tree のリーフノードに関連付けられるグラフのノードの最大数は 100 とし、インターナルノードの分岐数は 5 とした。つまり、インターナルノードの空間は vantage point を中心とした 4 つの異なる半径の超球により 5 つの部分空間に分割される。この構成の場合には、リーフノードの点の数が 100 を超えるとリーフノードを分割して 20 の点を有するリーフノードが 5 つ生成される。したがって、リーフノードに平均 60 の点が存在すると仮定できるので、10 万の点を格納するには約 1,667 のリーフノードが必要となる。木構造を 5 つに分割しているのでルートノードの層を除いて 5 階層あれば 3,125 のリーフノードが生成可能である。したがって、5 階層ですべての点を格納可能であり、つまりは、平均 5 回の距離計算でルートからリーフノードへ到達できる。ただし、vp-tree とは異なり dvp-tree はルートノードからリーフノードまでの枝の数が必ずしも一定とはならないので、実際の距離計算は平均 5 回を多少上回ると予想される。一方、20 次元のデータでエッジ数が 8 のときに ANNG の 1 段目の処理である近傍検索時に約 80 回の距離計算を行うことが予備実験から分かっている。したがって、この構成による dvp-tree を用いれば ANNG の近傍検索に比べると ANNGT はきわめて少ない距離計算回数となると予想できる。

範囲検索および k 最近傍検索の検索精度の測定では、各評価ごとに検索を 50 回試行して各指標の平均値を求めた。k 最近傍検索の検索結果の検索数は 20 とした。また、範囲検索では検索結果の検索数が k 最近傍検索と同一の 20 件となる検索半径をあらかじめ決定し検索を行った。これは、同一の検索数を保証することによって、範囲検索と k 最近傍検索の比較を可能とするためである。

アルゴリズム上、検索結果に誤検索は含まれず、適合率はつねに 1.0 となるので、検索精度の指標として再現率を用いた。以下に検索精度の指標である再現率と適合率の算出式を示す。なお、正答数は検索されるべき検索結果の数である。

$$\text{再現率} = \frac{\text{検索結果内の正答数}}{\text{全正答数}} \quad (3)$$

$$\text{適合率} = \frac{\text{検索結果内の正答数}}{\text{検索結果数}} \quad (4)$$

5.1 一様分布データによるインデックス生成の比較評価

グラフのエッジ数は検索精度と密接に関係するので、各インデックスを比較するにはインデックス全体のエッジ数をほぼ同数にする必要がある。kNNG では k 個の最近傍のノードへの有向エッジを生成するので、 n 個のノードを登録するとインデックス全体では kn 本の有向エッジが生成される。ANNG や ANNGT は生成時に検索数 j の k 最近傍検索により j 本の無向エッジが生成される。つまり、有向エッジで表現すると $2j$ 本の有向エッジが生成され、インデックス全体では $2jn$ 本の有向エッジが生成されることになる。つまり、kNNG のエッジ k 本に対して ANNG や ANNGT では無向エッジ $k/2$ 本を生成することにより、全体では、ほぼ同等数のエッジで構成されるグラフが生成されることになる。したがって、本評価ではたとえばエッジ数 k を 4 とする ANNG および ANNGT とは、生成時に検索数を 2 とする k 最近傍検索を行って生成した ANNG および ANNGT を意味する。

動的な追加登録を前提とすると kNNG のインデックス生成時にすべてのノード間の距離を求める必要があるので、ノード数に対して距離計算回数が一意的に $O(n^2)$ と決まる。一方、ANNG は探索半径係数の値により距離計算回数が変動する。筆者は ANNG の探索半径係数 0.1 の場合に ANNG のインデックス生成時の距離計算回数が kNNG の約 1.6% しかないことを示した¹⁾。本評価では ANNG および ANNGT について詳細に比較する。次元数 20、探索半径係数 0.1 において各エッジ数ごとのインデックス生成途中における 1 ノード登録時の距離計算回数を図 5 に示す。図中の anng および anngt に続く数字はエッジ数を示す。図より両者ともに累積登録数が多いほど 1 ノードの登録時における距離計算回数の増加傾向は抑えられており、大量のノードを登録できると予想される。また、ANNG に比べ ANNGT の方が距離計算回数が全体にわたって少ないことが示されており、ANNGT がインデックス生成において有効であることを確認した。

5.2 一様分布データによる検索の比較評価

ANNG および ANNGT のインデックス生成時には k 最近傍検索を行うので、検索の精度が異なるとインデックス生成の精度も異なる。そこで、検索の精度のみを評価するために、ANNGT で生成したインデックスに対して ANNG および ANNGT の検索精度を評価した。

比較の方法として、1 段目の処理のみを比較評価する方法も考えられるが、ANNG では、

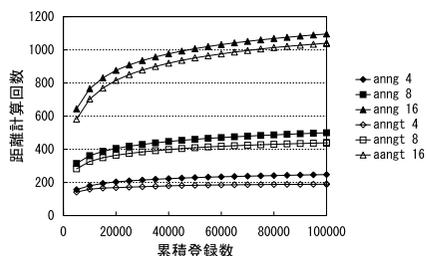


図5 エッジ数に対する累積登録数と1ノード登録時の距離計算回数の関係

Fig. 5 Total number of inserted nodes versus number of distance computations of each node insertion.

1 段目で単一の最近傍ノードだけを探索し、その単一のノードを2段目の探索の起点とする一方、ANNGTでは1段目で複数の近傍ノードを探索し、その複数のノードを2段目の探索の起点とする。したがって、1段目の処理の結果が2段目の処理の距離計算回数にも影響を与えるので、1段目および2段目の全処理を通しての距離計算回数を測定した。探索半径係数を0.0から0.1刻みで増やし、各探索半径係数における距離計算回数と検索精度を測定した。なお、検索精度が0.98を超えたところで測定を終了した。

エッジ数が16において一様分布データに対するANNGおよびANNGTの範囲検索時の距離計算回数に対する検索精度を図6に、k最近傍検索時の距離計算回数に対する検索精度を図7に示す。図よりほぼ全般にわたってわずかではあるがANNGTの検索精度がANNGより上回っており、ANNGTが有効であることを確認した。

次に、それぞれの検索方法によりインデックスを生成した場合の評価を行った。エッジ数 k が4, 8, 16において、一様分布データに対するkNNG, ANNGおよびANNGTの範囲検索時の距離計算回数に対する検索精度を図8に、k最近傍検索時の距離計算回数に対する検索精度を図9に示す。範囲検索の検索数をk最近傍検索の検索数と同数としていることもあり、範囲検索とk近傍検索ではほぼ同様の傾向を示している。kNNGはグラフの連結性を保証していないので、エッジ数が少ないほどたどれないノードが増加し、その結果距離計算回数が多くても検索精度が上がらない傾向がある。ANNGとANNGTを比較するとANNGよりANNGTが全般にわたって距離計算回数が少なく、ANNGTが有効であることを確認した。

さらに、エッジ数が4でのk最近傍検索において検索数を変化させた場合の距離計算回数に対する検索精度を図10に示す。図中のannngおよびanngtに続く数値が検索数であ

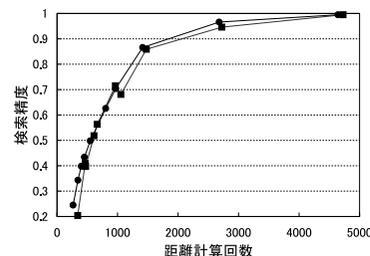


図6 ANNGTのインデックスを用いた範囲検索における距離計算回数と検索精度の関係

Fig. 6 Number of distance computations versus recall for range search using ANNGT index.

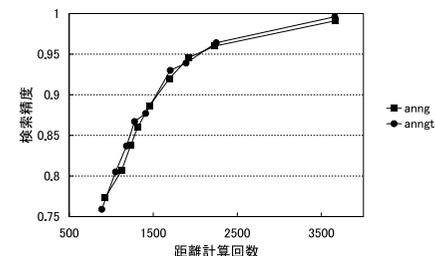


図7 ANNGTのインデックスを用いたk最近傍検索における距離計算回数と検索精度の関係

Fig. 7 Number of distance computations versus recall for nearest neighbor search using ANNGT index.

る。検索数が増加すると若干検索精度が低下するが、ANNGTの精度がいずれの検索数でもANNGを上回っており、検索数に関係なくANNGTが有効であることを確認した。図示しないがエッジ数が8および16の場合も同様の傾向を示した。

5.3 一様分布データによる既存のインデックスとの比較評価

一様分布データに対して既存の距離空間インデックスとの比較評価を行った。比較したインデックスはMetric Spaces Library^{*1}で提供されているLAESA, sa-tree,.mvp-treeである。k最近傍検索時の各インデックスの距離計算回数を表1に示す。mvp-treeに関しては木構造上のノードの分岐数が2と6の場合について測定した。木構造上の各ノードのvantage pointの数は1である。なお、mvp-treeのvantage pointとはvp-treeと同様に木構造を構成する各ノードの部分空間を分割するための基準となる点のことである。いずれのインデックスもほぼ全件の距離計算を行っている。なお、sa-treeのみ検索漏れが生じる可能性のある近似検索なので、検索精度が1.0に達していない。

LAESAについてはpivotを増やせば距離計算回数が減るので、pivot数に対する距離計算回数を測定し、その結果を図11に示す。なお、LAESAのpivotは距離計算の基準となるノードのことである。pivot数が5,000ぐらいまではpivot数が増加するにつれ距離計算回数が減少するが、それ以上になると逆に距離計算回数が増加する。pivot数が5,000のときに距離計算回数は最少である約20,000となる。ANNGおよびANNGTでは図9のよう

*1 http://www.sisap.org/Metric_Space_Library.html

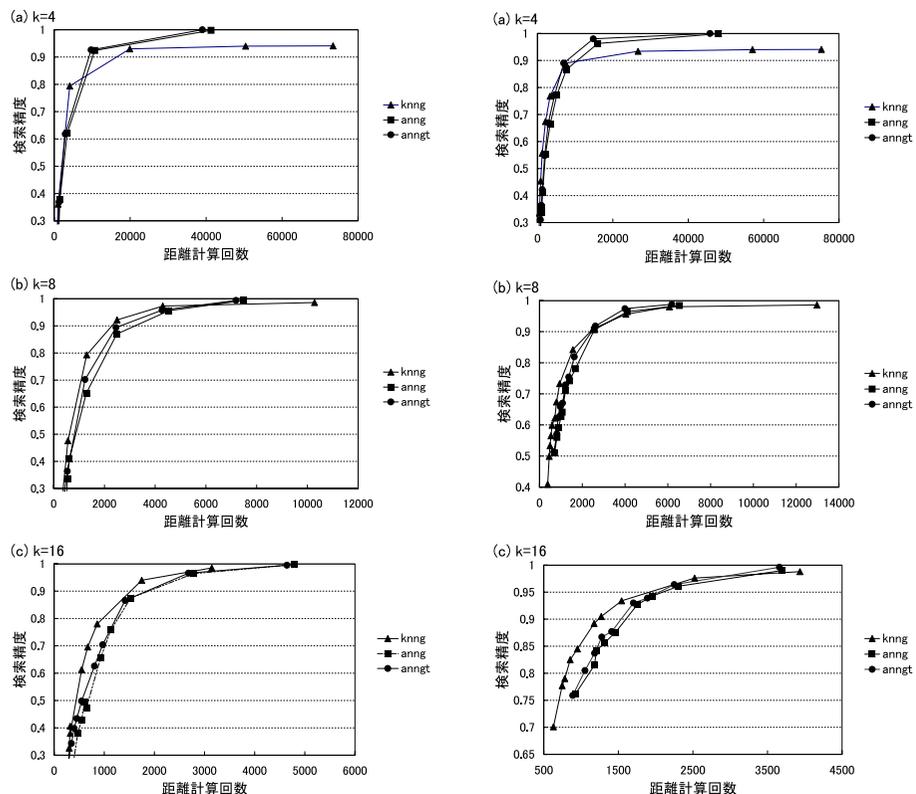


図 8 一様分布データの範囲検索における距離計算回数と検索精度の関係

Fig. 8 Number of distance computations versus recall for number of results by range search using uniform distribution data.

図 9 一様分布データの k 最近傍検索における距離計算回数と検索精度の関係

Fig. 9 Number of distance computations versus recall for k nearest neighbor search using uniform distribution data.

にエッジ数が 8 であれば検索精度 0.98, 距離計算回数は約 7,000 となり, LAESA より検索精度は若干低下するが, 距離計算回数は半分以下に削減できている.

以上の結果より, ANNG および ANNGT は近似検索ではあるものの, 既存の距離空間インデックスよりも検索時の距離計算回数に関して優位であることを確認した.

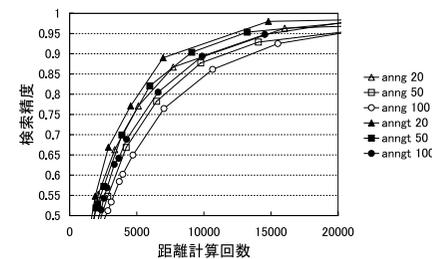


図 10 一様分布データの k 最近傍検索における検索数に対する距離計算回数と検索精度の関係

Fig. 10 Number of distance computations versus recall for number of results by k nearest neighbor search using uniform distribution data.

表 1 主な距離空間インデックスの距離計算回数と検索精度

Table 1 Number of distance computations and recall for principal metric indexes.

インデックス	LAESA	sa-tree	mvp 2	mvp 6
距離計算回数	89,706.4	91,532.3	100,000.0	100,000.0
検索精度	1.0	0.937	1.0	1.0

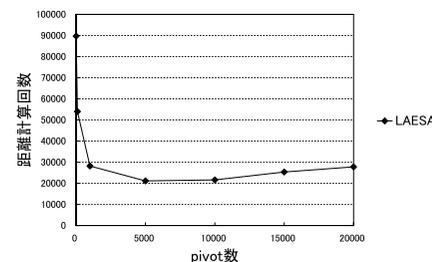


図 11 LAESA における pivot 数と距離計算回数の関係

Fig. 11 Number of pivots versus number of distance computations for number of dimensions using LAESA.

5.4 画像特徴量による比較評価

エッジ数 k が 4, 8, 16 において, 画像特徴量による範囲検索の結果を図 12 に, k 最近傍検索の結果を図 13 に示す. 利用した画像特徴量は 1,000 次元を超えるが一様分布のデータと比較すると検索精度がかなり向上している. 画像特徴量では次元数が多くても特徴量空間上で分布に偏りがあり, インデックスの効果が高いと考えられる. 評価に用いたデータの各次元が 1 バイトの整数型に量子化されていることも一因と考えられる. 量子化されてい

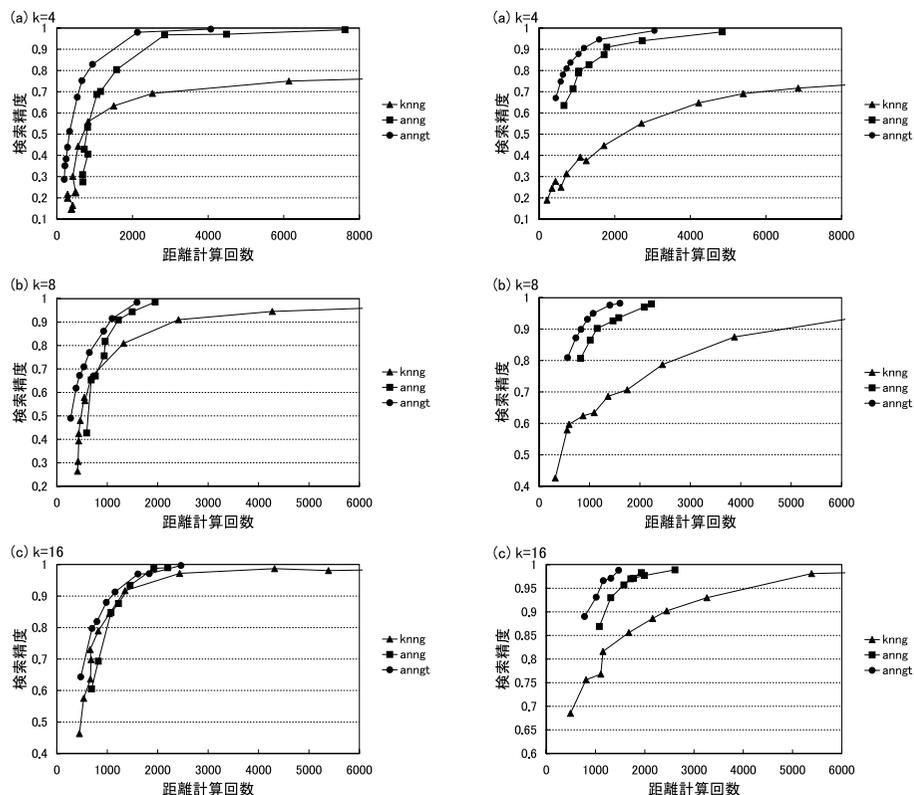


図 12 画像特徴量の範囲検索における距離計算回数と検索精度の関係

Fig. 12 Number of distance computations versus recall for range search using image features.

ことで、任意の 2 つの特徴量の対応する要素を個別に比べた場合に完全一致する要素数が量子化前に比べてきわめて多くなる。一致することで距離関数としてはその次元は無視されるに等しく、一致する要素が多いほど結果的に次元数が削減されるのと同様の効果があると予想する。これらの結果から、ANNGT は実データである画像特徴量の検索においても ANNG よりも有効であることを確認した。

図 13 画像特徴量の k 最近傍検索における距離計算回数と検索精度の関係

Fig. 13 Number of distance computations versus recall for k nearest neighbor search using image features.

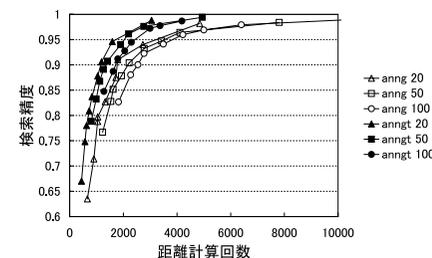


図 14 画像特徴量の k 最近傍検索における検索数に対する距離計算回数と検索精度の関係

Fig. 14 Number of distance computations versus recall for number of results by k nearest neighbor search using image features.

また、範囲検索より k 最近傍検索の方がより ANNGT の効果があることが確認できる。k 最近傍検索では、探索範囲の絞り込みが速いほど距離計算回数を削減できる。ANNGT では vp-tree により得られるノード数が平均 60 であり、検索数 20 より多いので、2 段目のグラフの探索開始時にすでに探索半径が限定できる。ANNG では、グラフ探索開始時には 1 ノードしか獲得できていないので、探索半径が限定できない。一方、範囲検索では、探索半径が確定しているので、探索範囲の絞り込みの効果が得られない。このことから、範囲検索より k 最近傍検索の方がより ANNGT の効果が表れていると判断する。

エッジ数が 4 での k 最近傍検索において検索数を変化させた場合の距離計算回数に対する検索精度を図 14 に示す。図中の annng および anngt に続く数値が検索数である。一様分布データの場合と同様に検索数が増加すると若干検索精度が低下するが、ANNGT の精度がいずれの検索数でも ANNG を上回っており、検索数に関わらず ANNGT が有効であることを確認した。図示しないが一様分布データの場合と同様にエッジ数が 8 および 16 の場合も同様の傾向を示した。

6. おわりに

多次元データの最近傍検索を実現するグラフ構造型のインデックスとして ANNG のコスト（同一精度における距離計算回数）の削減に対する有効性が確認されている。ANNG では、ランダムなノードから検索点に対して近傍ノードを探索する処理、その近傍ノードを起点として探索範囲内のノードを網羅的に探索する処理、の 2 段階の処理からなる。しかし、1 段目の処理ではランダムなノードから近傍ノードを探索するので、ノード数の増加に比例してコストが増大するという課題がある。そこで、近傍ノードの探索用のインデックスを用

いて, 1 段目のコストを抑制できると考えた. ただし, インデックスには以下の条件を満たす必要がある.

- 距離空間インデックスである.
- 次元の呪いに耐性がある.
- データの追加が可能な動的なインデックスである.

そこで, これらの条件を満たす *dvp-tree* を利用して 1 段目の探索を行う ANNGT を提案し, 一様分布データおよび画像特徴量を用い ANNG よりもインデックス生成や検索のコストが低減できることを確認した.

ANNG の 2 段目の探索はグラフの局所的な探索にすぎないので, ノード数が多くなっても, コストの増加は比較的少ない. さらに, ANNGT により 1 段目の探索もノード数に比較的影響を受けにくくなったので, 大規模なデータへの適用が可能となった. そこで今後は大規模なデータに ANNGT を適用しどのような挙動を示すかを検証する予定である.

参 考 文 献

- 1) 岩崎雅二郎: 近似 k 最近傍グラフによる距離空間の近傍検索, 情報処理学会論文誌 データベース, Vol.3, No.1(TOD45), pp.18–28 (2010).
- 2) Guttman, A.: R-trees: A dynamic index structure for spatial searching, *ACM Sigmod Record*, Vol.14, No.2, pp.47–57 (1984).
- 3) Beckmann, N., Kriegel, H., Schneider, R. and Seeger, B.: The R*-tree: An efficient and robust access method for points and rectangles, *ACM SIGMOD Record*, Vol.19, No.2, pp.322–331 (1990).
- 4) Bentley, J.: Multidimensional binary search trees used for associative searching, *Comm. ACM*, Vol.18, pp.509–517 (1975).
- 5) Samet, H.: The quadtree and related hierarchical data structures, *ACM Computing Surveys CSUR*, Vol.16, No.2, pp.187–260 (1984).
- 6) White, D. and Jain, R.: Similarity indexing with the SS-tree, *Proc. 12th International Conference on Data Engineering*, pp.516–523 (1996).
- 7) Kurniawati, R., Jin, J. and Shepherd, J.: The SS⁺-tree: An improved index structure for similarity searches in a high-dimensional feature space, *Proc. SPIE/IS&T Conf. on Storage and Retrieval for Image and Video Databases V*, Vol.3022, pp.110–120 (1997).
- 8) Berchtold, S., Keim, D. and Kriegel, H.: The X-tree: An index structure for high-dimensional data, *Readings in Multimedia Computing and Networking*, p.451 (2001).
- 9) Weber, R., Schek, H. and Blott, S.: A quantitative analysis and performance study

- for similarity-search methods in high-dimensional spaces, *Proc. International Conference on Very Large Data Bases*, IEEE, pp.194–205 (1998).
- 10) Arya, S., Mount, D., Netanyahu, N., Silverman, R. and Wu, A.: An optimal algorithm for approximate nearest neighbor searching fixed dimensions, *J. ACM*, Vol.45, No.6, pp.891–923 (1998).
 - 11) Gionis, A., Indyk, P. and Motwani, R.: Similarity search in high dimensions via hashing, *Proc. 25th Internat. Conf. on Very Large Data Bases*, pp.518–528 (1999).
 - 12) Ioka, M.: A method of defining the similarity of images on the basis of color information, IBM Res., Tokyo Res. Lab., Technical Report RT-0030 (1989).
 - 13) Faloutsos, C., Barber, R., Flickner, M., Hafner, J., Niblack, W., Petkovic, D. and Equitz, W.: Efficient and effective querying by image content, *Journal of Intelligent Information Systems*, Vol.3, No.3, pp.231–262 (1994).
 - 14) Seidl, T. and Kriegel, H.: Efficient user-adaptable similarity search in large multimedia databases, *Proc. International Conference on Very Large Data Bases*, IEEE, pp.506–515 (1997).
 - 15) Uhlmann, J.: Satisfying general proximity/similarity queries with metric trees, *Information Processing Letters*, Vol.40, No.4, pp.175–179 (1991).
 - 16) Brin, S.: Near neighbor search in large metric spaces, *Proc. International Conference on Very Large Data Bases*, IEEE, pp.574–584 (1995).
 - 17) Yianilos, P.: Data structures and algorithms for nearest neighbor search in general metric spaces, *Proc. 4th Annual ACM-SIAM Symposium on Discrete Algorithms*, Society for Industrial and Applied Mathematics Philadelphia, PA, USA, pp.311–321 (1993).
 - 18) Bozkaya, T. and Ozsoyoglu, M.: Distance-based indexing for high-dimensional metric spaces, *Proc. 1997 ACM SIGMOD International Conference on Management of Data*, ACM New York, NY, USA, pp.357–368 (1997).
 - 19) Ciaccia, P., Patella, M. and Zezula, P.: M-tree: An efficient access method for similarity search in metric spaces, *Proc. International Conference on Very Large Data Bases*, Citeseer, pp.426–435 (1997).
 - 20) 岩崎雅二郎: 類似画像検索を実現する距離空間インデックスの実装及び評価, 情報処理学会論文誌 データベース, Vol.40, No.SIG3(TOD1), pp.24–33 (1999).
 - 21) Ruiz, E.: An algorithm for finding nearest neighbours in (approximately) constant average time, *Pattern Recognition Letters*, Vol.4, No.3, pp.145–157 (1986).
 - 22) Micó, M., Oncina, J. and Vidal, E.: A new version of the nearest-neighbour approximating and eliminating search algorithm (AESAs) with linear preprocessing time and memory requirements, *Pattern Recognition Letters*, Vol.15, No.1, pp.9–17 (1994).
 - 23) Okabe, A., Boots, B. and Sugihara, K.: *Spatial tessellations: Concepts and appli-*

- cations of Voronoi diagrams*, John Wiley & Sons, Inc. New York, NY, USA (1992).
- 24) Navarro, G.: Searching in metric spaces by spatial approximation, *The VLDB Journal*, Vol.11, No.1, pp.28–46 (2002).
- 25) Sakagaito, J. and Wada, T.: Nearest first traversing graph for simultaneous object tracking and recognition, *IEEE Conference on Computer Vision and Pattern Recognition, 2007. CVPR'07*, pp.1–7 (2007).
- 26) Sebastian, T. and Kimia, B.: Metric-based shape retrieval in large databases, *Proc. 16th International Conference on Pattern Recognition*, Vol.3, pp.291–296 (2002).
- 27) Callahan, P. and Kosaraju, S.: A decomposition of multidimensional point sets with applications to k-nearest-neighbors and n-body potential fields, *J. ACM*, Vol.42, No.1, pp.67–90 (1995).
- 28) Chen, J., Fang, H. and Saad, Y.: Fast Approximate kNN Graph Construction for High Dimensional Data via Recursive Lanczos Bisection, *J. Machine Learning Research*, Vol.10, pp.1989–2012 (2009).

- 29) Plaku, E. and Kavragi, L.: Distributed computation of the knn graph for large high-dimensional point sets, *J. Parallel and Distributed Computing*, Vol.67, No.3, pp.346–359 (2007).

(平成 22 年 4 月 27 日受付)

(平成 22 年 11 月 5 日採録)



岩崎雅二郎 (正会員)

1987 年早稲田大学工学部工業経営学科卒業。1989 年同大学院理工学研究科機械工学専攻修士課程修了。同年日本電気 (株) 入社。1990 年 (株) リコー入社。全文検索, 画像検索, 文書検索の研究開発に従事。2007 年 ヤフー (株) 入社。画像検索の研究開発に従事。