社団法人　情報処理学会　研究報告
IPSJ SIG Technical Report

2006－CVIM－153（7）
2006／3／16

空間分割と直交変換の統合による高次元最近傍探索の高速化

荒井 英剛† 加藤 丈和† 和田 俊和†

†和歌山大学 システム工学部 システム工学研究科
〒 640-8510 和歌山県 和歌山市 栄谷 930

E-mail : araih@vrl.sys.wakayama-u.ac.jp,t.kato@ieee.org,twada@ieee.org

あ ら ま し 事例に基づく学習や認識において，最近傍探索技術は重要な基礎技術として位置づけられる．従来，最近傍探索の高速化技術として様々な手法が提案されているが，これらのほとんどの手法は最近傍候補の絞り込みによる手法である．しかし，30 次元を超える高次元空間ではこれらの手法はほとんど全探索となり，画像を直接扱うような，数千から数万次元で有効な最近傍探索の高速化手法は提案されていない．本研究では，K-D Tree や ANN で採用されている空間分割に基づく絞り込み手法に，主成分分析による直交変換を統合することで，高次元でも有効な最近傍探索の高速化手法を提案する．

キーワード 高次元空間における最近傍探索，空間分割，ANN，主成分分析

Nearest Neighbor Search in High-Dimensional Space based on Space Decomposition and Orthogonal Basis Selection

Hidetaka Arai†, Takekazu Kato†, Toshikazu Wada†

†Graduate School of Systems Engineering, Wakayama University,
930 Sakaedani, Wakayama, 640-8510, Japan

E-mail : araih@vrl.sys.wakayama-u.ac.jp, t.kato@ieee.org, twada@ieee.org

Abstract Nearest Neighbor (NN) search is essential for case/instance based reasoning. Most NN search methods proposed so far are based on NN candidate narrowing and these methods degenerates to brute force search in high-dimensional space over 30D. Efficient NN search algorithm dealing with image database in over thousand dimensional space has not been proposed. In this paper, we propose an effective NN search algorithm integrating orthogonal expansion by PCA and space decomposition, which works in very high dimensional space. This method can be regarded as a natural extension of ANN. Through extensive experiments, we confirmed the efficiency of our method.

Keywords Nearest Neighbor Search in high-dimensional space，space decomposition，ANN，PCA

1 まえがき

本研究の目的は，大量かつ高次元データに対する最近傍探索の高速化である．最近傍探索は，入力パターンが与えられたときに，あらかじめ記憶していたプロトタイプの集合から，最も距離が近いパターンを探索する技術であり，パターン認識，データマイニングなどに応用可能な重要な基礎技術である．また，近年，これらの分野では，画像認識や類似画像検索など，大量の高次元データを高速に扱う技術が望まれている．

一般に，全てのプロトタイプ集合に対して入力パターンとの距離を計算し，比較すると次元数とプロトタイプ数に比例した計算コストがかかるため，高次元かつ大量のデータに対する探索は時間がかかる．従来提案されてきた最近傍探索の高速化手法は，プロトタイプ集合から最近傍となる候補を絞り込むことによって，実際に入力パターンとの距離を比較するプロトタイプの数を削減する手法 ([3]，[6]，[5]，[4]) と，SSDA(Sequential Similarity Detection Algorithm)[9] と呼ばれる，距離計算の打ち切りによって，平均的な距離計算コストを削減する手法に分けることができる．

前者の手法は，絞り込みのための様々な方法が提案されているが，次元が大きくなるほどその効果が減少し，いずれも 30 次元程度からほぼ全探索に近くなっ

てしまう．そのため画像のような数千から数万次元の高次元データに対しては，効果が期待できない．一方で後者の距離計算の打ち切りは，低次元では大きな効果はないものの，高次元でもその効果が薄れることなく，ある程度の高速化が期待できる．

これらのアプローチを組み合せた手法も提案されている．Approximate Nearest Neighbor(ANN)[1] は，kd-tree[3] などと同様の空間分割による最近傍候補の絞り込みと，最近傍候補との距離を比較するときに距離計算の打ち切りを行っている．ANN は一般的な分布のプロトタイプ集合に対して最速な最近傍探索アルゴリズムであり，高次元データに対しても比較的高速であるが，特に高次元データを意識したアルゴリズムでないため，高次元で効率的であるとは言えない部分もある．

そこで本研究では，ANN のアルゴリズムの利点と弱点について分析し，利点をいかしつつ弱点を改善し，高次元データに対しても効率の良い最近傍探索手法を提案する．

以下，2 章では ANN を含む，従来の最近傍探索手法の概略を説明する．また 3 章では ANN の弱点を分析した上で，それを改善する方法を提案する．4 章では顔画像データによる実験結果を示し，5 章でまとめと今後の課題について延べる．

2 問題定義と関連研究

本章では，まず最近傍探索の問題定義を行った上で，従来の最近傍探索の高速化手法の概要を紹介し，ANN の具体的なアルゴリズムについて説明する．

2.1 問題定義

最近傍探索とは，プロトタイプと呼ばれる学習パターン集合 $\mathcal{P} = \{\boldsymbol{p}_1, \cdots, \boldsymbol{p}_n\}$ が与えられているときに，入力パターン $\boldsymbol{q}$ との距離が最も小さいパターン $\boldsymbol{p}^*$ を求めることである．

二つのパターン $\boldsymbol{p}, \boldsymbol{q}$ 間に距離 $d(\boldsymbol{p}, \boldsymbol{q})$ が定義されているとき，最近傍探索は次式のように表わされる．

$$\boldsymbol{p}^* = \arg\min_{\boldsymbol{p} \in \mathcal{P}} d(\boldsymbol{q}, \boldsymbol{p}) \tag{1}$$

距離 d の定義は一般的には，距離の公理 (対称性，非負性，三角不等式) を満していれば問題に応じて適切に設定すれば良いが，パターンが d 次元のベクトル空間で表されている場合には L_n 距離が用いられることが多く，また最近傍探索アルゴリズムも L_n 距離を前提としているものが多い．

$$d(\boldsymbol{p}, \boldsymbol{q}) = L_n(\boldsymbol{p}, \boldsymbol{q}) = \sqrt[n]{\sum_i |x_i - q_i|^n} \tag{2}$$

この場合，d 次元のパターン間の距離を計算するのに，d に比例する計算コストが必要であり，また，N 個のプロトタイプの中から全探索によって最近傍パターンを探索する場合の計算コストは $O(dN)$ となり，高次元になるほど，またプロトタイプ数が大量になるほど，探索に計算時間がかかる．

2.2 従来の高速化手法

従来提案されている最近傍探索の高速化手法は，最近傍候補の絞り込みによるアプローチと距離計算の打ち切りによるアプローチの 2 種類に大別できる．

最近傍候補の絞り込み

このアプローチは，最近傍となる可能性のあるプロトタイプの候補を絞り込み，入力パターンとの距離計算を行うプロトタイプの数を削減することで，高速化を図っている．このアプローチに基づく手法には，プロトタイプの分布に基いてパターン空間をいくつかの領域に分割しておき，最近傍候補が含まれる可能性のある領域を絞り込んでおく方法 (ANN[1], GNAT[6], VPtree[5]) や，あらかじめプロトタイプ同士の距離を計算しておき，三角不等式に基いて候補を絞り込む手法 (LAESA[4],VPtree[5]) などが提案されている．

これらの手法では，比較的低次元の場合には効果的であるが，パターン空間が 30 次元程度以上になると，ほぼ全探索となり絞り込みの効果がなくなっていく．

図 1 に最近傍候補の絞り込みに基づく手法 (LAESA,GNAT,VPtree) について，2 次元から 30 次元のデータに対して最近傍探索を行った場合の距離計算を行ったプロトタイプ数を示す．この結果より，どの手法も 30 次元ではほぼ全探索となってしまい絞り込みの効果はなくなっていることがわかる．

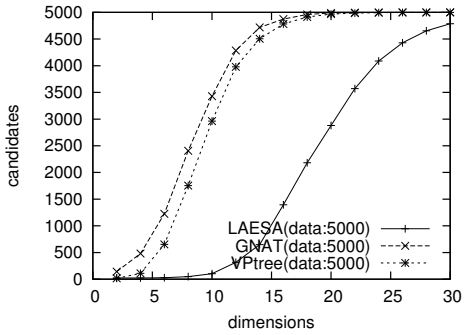


図 1: 次元の変化に対する距離計算回数

距離計算の打ち切り

このアプローチでは，各プロトタイプとの距離計算を，最近傍となりえないと判明した時点で打ち切り，平均的な距離計算コストを削減することで高速化を図っている．

この手法は，SSDA(Sequential Similarity Detection Algorithm) とも呼ばれ，低次元では絞り込み手法に比べて効果が低いが，高次元になっても効果的である．

パターン $\boldsymbol{x}, \boldsymbol{y}$ 間の距離を以下のような L_m^m とする．

$$d(\boldsymbol{x}, \boldsymbol{y})^m = \sum_{i=1}^{d} (\boldsymbol{x}_i - \boldsymbol{y}_i)^m \qquad (3)$$

このとき，次元数 d よりも小さい次元 $b(1 \leq b < d)$ までの基底を用いて次式のように下限値を計算できる．

$$\text{下限値：} \quad \underline{d}(\boldsymbol{x}, \boldsymbol{y})^m = \sum_{i=1}^{b} (\boldsymbol{x}_i - \boldsymbol{y}_i)^m \qquad (4)$$

この下限値 $\underline{d}$ は基底数 b の増加に伴い単調に増加し，距離 d よりも小さく，次式が成り立つ．

$$\underline{d}(\boldsymbol{x}, \boldsymbol{y})^m \leq d(\boldsymbol{x}, \boldsymbol{y})^m \qquad (5)$$

つまり，基底数 b を一つずつ増やしながら，入力パターン $\boldsymbol{q}$ とプロトタイプ $\boldsymbol{p}$ との距離の下限値 $\underline{d}(\boldsymbol{q}, \boldsymbol{p})^m$ を計算し，これが暫定値 r より大きくなったところで，$\boldsymbol{p}$ は最近傍解とはなりえないことが分るので，そこで距離計算を打ち切ることができる．

ANN では，最近傍候補の絞り込みに加え距離計算の打ち切りも同時に行っている．また，PCA などの直交変換を行って寄与率の高い基底から順に距離計算を行うことによって，より効果的に打ち切りが行えるという報告 [8] もされている．

図 2 に前節の 4 つの手法に加え，SSDA と ANN による最近傍探索の時間を示す，この結果からも高次元では距離計算の打ち切りを用いる SSDA や ANN が比較的高速であることが分かる．

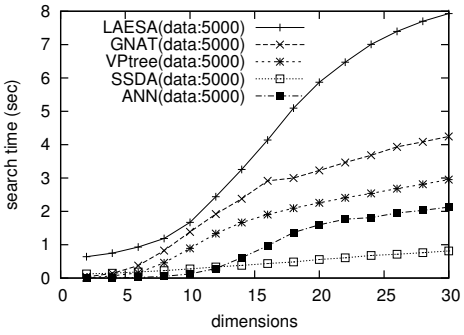
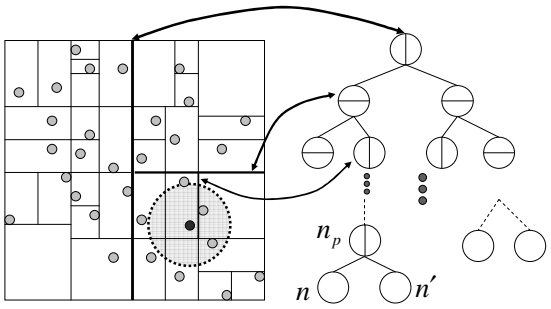


図 2: 手法ごとの最近傍探索時間

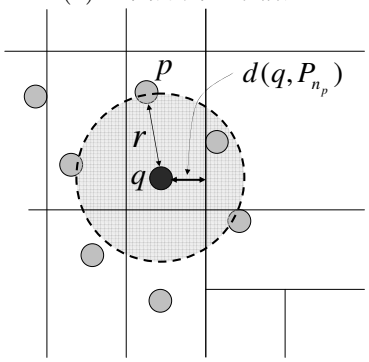
2.3 Approximate Nearest Neighbor

本節では，Approximate Nearest Neighbor(ANN) の概要について説明する．ANN は，パターン空間を分割し，最近傍パターンを含む可能性がある領域を最近傍候補として絞り込みを行う探索手法であるとともに，距離計算の打ち切りも行うことで高次元でも比較的高速な探索を実現している．

パターン空間の分割は，図 3 に示すように，パターン空間の各軸に対して，プロトタイプが分布する幅を調べ，分布幅が最大となる軸（**分割軸**）に直交な平面で分割（**分割面**）する．さらに分割された各領域についても分布幅が最大となる軸に直交な平面で分割して



(a) 空間分割と木構造



(b) 拡大図

図 3: ANN による空間分割と木構造

いき，各領域に含まれるパターンが 1 個となるまで再帰的に分割を繰り返す．またこの再帰的な分割は各分割面をノードとし，各プロトタイプを含む領域をリーフとする 2 分木に対応付けて表現する．

入力パターンが与えられた時には，入力パターンが含まれる領域の探索，最近傍パターンを含む可能性のある領域の絞り込み，候補領域内のプロトタイプとの距離計算の 3 つの手順によって，最近傍パターンを探索する．このそれぞれの手順について詳しく説明する．

1. 入力パターンを含む領域の探索

   入力パターンを含む領域の探索は，2 分木探索によって行う．図 3 に示すように 2 分木のルートノードから，2 分木の構造に従って各ノードに対応付けられた分割面のどちらに含まれるかを再帰的に判定する．このとき各分割面との比較は分割軸の成分の比較であり，これは特徴ベクトルのある一つの要素に関する閾値処理を行うだけである．

   こうして探索した，入力パターンを含む領域中のプロトタイプを暫定解 $\boldsymbol{p}$ とし，暫定解と入力パターンとの距離を暫定値 r とする．

2. 最近傍候補を含む領域の探索

   1 で求めた暫定解は，入力パターンと同じ領域に含まれるが，図 3 に示すようにこの暫定解が最近傍解とは限らない．そこで，2 分木を逆に辿りながら最近傍階を含む領域を探索する．

   暫定解を含むリーフノードを n，その親ノードを n_p，兄弟ノードを n' とし，ノード n_p に対応付

けられた分割面を P_{n_p} とする．入力パターンと分割面の距離 $d(\boldsymbol{q}, P_{n_p})$ が暫定値 r より小さいとき，この分割面の入力パターンを含まない領域に最近傍解が存在する可能性がある．このとき，兄弟ノード n' から再び 2 分探索を行い，n' 以下で入力パターンと最も近い領域を探索する．探索された領域に含まれるプロトタイプと入力パターンとの距離を計算し，それが暫定値 r 以下なら，暫定解と暫定値を更新し，同様の処理を続ける．

親ノードの分割面との距離 $d(\boldsymbol{q}, P_{n_p})$ が暫定値 r より大きい場合，つまり兄弟ノードの領域に最近傍解が含まれる可能性がない場合には，さらにその親ノードを辿って処理を続け，ルートノードに至るまで処理を続ける．

このような処理によって最近傍解が含まれる可能性のある領域を効率良く求めることができ，最小限のプロトタイプとの距離計算のみによって，最近傍解を探索することができる．

3. リーフに含まれるプロトタイプとの距離計算

   ANN では，最近傍候補の絞り込みだけではなく，リーフに含まれるプロトタイプとの距離計算を行う際に，SSDA（残差逐次検定法：Sequential Similarity Detection Algrithm）と同様の距離計算の打ち切りを行うことによって，効率良く距離の比較を行う．

ANN が高速な理由として次の 3 つが挙げられる．

1. 2 分探索による暫定解の高速な探索．
2. 最近傍解が含まれる領域の候補の絞り込みにより距離計算を行うプロトタイプ数の削減
3. 距離計算の打ち切りによる平均的な距離計算コストの削減

3 再帰的直交変換による ANN の高速化

図 2 の予備実験結果に示すように，ANN は高次元空間においても他手法よりも比較的高速に最近傍探索を行うことができるが，本来，高次元空間での探索について考慮された方法ではないため，いくつかの問題点がある．そこで本研究ではこの ANN の問題点について考察し，高次元でも効率良く探索する方法について議論する．

3.1 ANN の問題点とその改善法

3.1.1 最近傍候補の絞り込みの問題

ANN も他の最近傍候補の絞り込みに基づく手法と同様に，高次元空間ではほぼ全探索となってしまい絞り込みの効果はなくなってしまう．これは，高次元になると入力とプロトタイプの距離の絶対値に対して，プロトタイプ間の相対的な距離の差が小さくなっていくことに起因する．

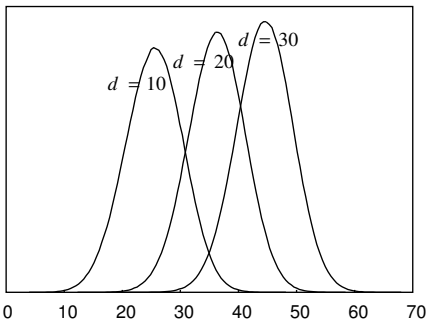


図 4: 次元の変化に対する距離の分布

図 4 に 10 次元，20 次元，30 次元の一様分布で生成したパターン間の相対的な距離のヒストグラムを示す．この結果からもパターン間の絶対的な距離に対するばらつきが小さくなっていることが明らかである．このような状態では，暫定解との距離を使っても最近傍候補を効率良く絞り込むことができなくなる．

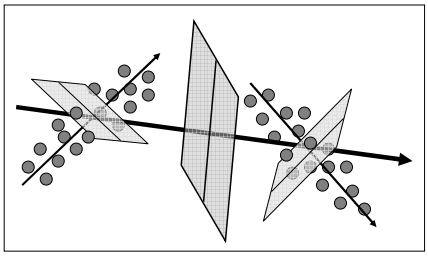


図 5: 第一主成分に基づく空間分割

実際には，データは一様分布ではなく偏った分布をしていることが多い．つまり，この偏りを分析し，分散が大きい軸によってパターン空間を分割することができれば，プロトタイプ間の距離の差が大きくなり，絞り込みの効率を改善できると考えられる．

このことを確認するための予備実験を行った．図 5 に示すように ANN の各ノードに対応する分割面を，領域に含まれるデータを主成分分析して得られる第一固有ベクトルに直交な平面で分割するように変更し，あとは ANN と同様に探索を行なった．この方法では各領域に含まれるパターン集合の分散が最大になる方向で分割しているため，最も効率の良い絞り込みが行えると考えられる．

表 1 は，顔画像集合に対してオリジナル ANN と，常に第一主成分で分割した場合の，距離計算を行ったプロトタイプの数を示している．この結果からオリジナル ANN ではほぼ全探索となっていたのが，第一主成分で分割した場合には，絞り込みの効果が表われていることが分かる．

手法	距離計算回数
オリジナル ANN	1000
常に第一主成分で分割	816

表 1: 距離計算回数の比較
1024 次元，プロトタイプ数は 1000

3.1.2 距離計算の効率の問題

もう 1 つの問題点として距離計算の途中打ち切りの効率が挙げられる．ANN では，プロトタイプとの距離計算を行う際に SSDA による距離計算の打ち切りを行っているが，この際にパターン空間の各基底に対して，どのような順序で距離計算を行うかについては考慮していない．

距離計算の打ち切りの効果を向上するためには，プロトタイプ間の距離をよく表現している軸から順に距離計算を行なうと良い．つまり，プロトタイプを主成分分析して得られる固有ベクトルを用いて，プロトタイプや入力パターンを直交変換しておいてから，寄与率の大きい軸の順に距離計算を行うことで効率良く距離計算の打ち切りを行うことが可能であると考えられる．

このことを確認するために，オリジナルの ANN と，あらかじめ固有空間への直交変換を行ってから ANN を適用した場合について，距離計算の効率を比較した．表 2 は，1024 次元のデータに対して距離計算に用いた基底数の平均を示している．オリジナルの ANN に比べて，あらかじめ直交変換しておいた場合では格段に距離計算効率が向上していることがわかる．

手法	距離計算時に用いた次元数
オリジナル ANN	413
あらかじめ直交変換	198

表 2: 距離計算効率の比較

3.2 本研究の基本アイデア

上記の ANN の問題点とその改善策に関する考察より，最近傍候補の絞り込みの効率を上げるためには，パターン空間の分割の際に，各領域に含まれるパターンの分散が最大になる軸，つまり第一主成分を分割軸として分割するのが良く，距離計算の打ち切りの効率を上げるためには，全てのパターンをプロトタイプの固有空間に直交変換しておき，寄与率の高い順に距離計算を行っていくと良いことが分かった．z そこで本研究では，これらの最近傍候補の絞り込みの効率と，距離計算の打ち切りの効率を同時に上げるために，これらの二つの効果を兼ね備えた手法を提案する．基本的なアイデアとしては，ルートノードからリーフノードまでのパス上のノードの各分割軸が直交し，かつ各ノードの領域内のプロトタイプの分散が最大となる軸，つまり第一主成分を分割軸として分割面を決定する．これによって，各分割軸によって効率の良い絞り込みを行いつつ，リーフノードまでの分割軸が直交しているのでそれらの軸を基底として距離計算の打ち切りが可能となる．

また，この方法では各ノードの分割軸との内積計算が必要となり，その内積計算のコストのために全体の探索速度は低下するが，この問題を解決するために，分割軸の個数を最小限に保つ方法を提案する．

3.3 提案手法の概要

提案手法では ANN と異なり，分割軸として常に分散が最大の軸を用いる．この空間分割に最適な分割軸は，空間を分割するごとに主成分分析を行い，常に分散が最大となる軸を求めることで得ることができる．

しかし，この場合分割軸同士の直交性が保たれないという問題が発生してしまう．そこで，分割後のデータを分割面に射影し，その分割面内で次の分割軸を決定する．分割面は分割軸に対する直交補空間なので，分割軸同士の直交性を保ちつつ最適な分割軸を決定することができる．

以下に提案手法による空間分割の具体的なアルゴリズムを説明する．また図 6 に本手法の概念図を示す．

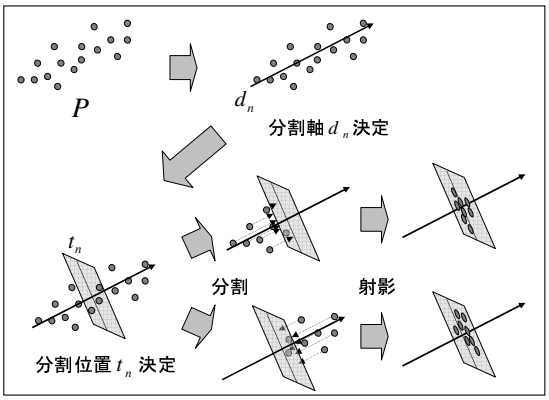


図 6: 提案手法の空間分割の流れ

1. 全プロトタイプ集合を $P = \{\boldsymbol{p}_1, \cdots, \boldsymbol{p}_n\}$ とし，二分木のルートノードを n とする．
2. プロトタイプ集合 P に対して主成分分析を行い第一主成分を分割軸 $\boldsymbol{d}_n$ とする．
3. 分割軸上のプロトタイプの重心位置を分割位置 t_n とし，分割軸とそれぞれのデータの内積を計算し，分割位置との大小によってプロトタイプを分割する．

$$P_l = \{\boldsymbol{p} \in P | \boldsymbol{p}^T \boldsymbol{d}_n < t_n\} \quad (6)$$
$$P_r = \{\boldsymbol{p} \in P | \boldsymbol{p}^T \boldsymbol{d}_n < t_n\} \quad (7)$$

4. それぞれのプロトタイプを分割面に射影する．

$$P_l^{'} = \{\boldsymbol{p}^{'} | \boldsymbol{p}^{'} = \boldsymbol{p} - \boldsymbol{p}^T \boldsymbol{d}_n \boldsymbol{d}_n,\ (\boldsymbol{p} \in P_l)\} \quad (8)$$

$$P_r^{'} = \{\boldsymbol{p}^{'} | \boldsymbol{p}^{'} = \boldsymbol{p} - \boldsymbol{p}^T \boldsymbol{d}_n \boldsymbol{d}_n,\ (\boldsymbol{p} \in P_r)\} \quad (9)$$

5. $P = P_l^{'}$ として，2から5を再帰的に繰り返し，それを左子ノードとする．また同様に $P = P_r^{'}$ として，2から5を再帰的に繰り返し，それを右子ノードとする．

以上の処理を各ノードにおけるプロトタイプが1個になるまで再帰的に繰り返して2分木を構成する．

探索方法は，基本的にはANNと同様であるが，各ノードでの判定の際に，分割軸と入力パターンとの内積を求め，その値と分割位置を比較することで探索する．2分木の探索とプロトタイプとの距離計算について具体的なアルゴリズムを以下に示す．

1. 入力パターンを $\boldsymbol{q}$ とする．また，探索ノード s をルートノードとし，$\boldsymbol{u} = ()^T$ とする．
2. 入力パターン $\boldsymbol{q}$ と分割軸 $\boldsymbol{d}_n$ との内積を求め，$\boldsymbol{u}$ に追加する．

$$r = \boldsymbol{q}^T \boldsymbol{d}_n, \boldsymbol{u} = (\boldsymbol{u}|r)^T \quad (10)$$

3. $r < d_n$ なら s を左子ノードに更新，そうでなければ右子ノードに更新
4. s がリーフノードならば探索を終了しプロトタイプとの距離計算を行い．そうでなければ，2へ．

また，一旦入力パターンが含まれる領域を探索した後に木を逆に辿りながら最近傍解が含まれる領域の候補を調べる方法はANNと同様に行う．

ここで，プロトタイプとの距離計算を行う際に，リーフノードまでのパス上の分割軸との内積値である $\boldsymbol{u} = u_1, \cdots, u_l$ を使って下限値を計算することができる．

$$\underline{d}^{'}(\boldsymbol{q}, \boldsymbol{p})^m = sum_{i=1}^{l} |u_i - q_i|^m \quad (11)$$

既に暫定解が選択され，暫定値 r が計算されている場合には，この $\underline{d}^{'}$ を r と比較して距離計算の打ち切りを行い，$\underline{d}^{'}$ が r より小さい場合のみプロトタイプとの距離計算を行う．

3.4 分割軸の再利用による内積回数の削減

前節で示した手法では，2分木を探索する際に，各ノード毎に入力パターンと分割軸との内積を計算する必要がある．そのため，内積計算の回数が増加し，全体の探索速度が低下してしまう．内積計算は次元数回の積和計算であり，これは，ほぼ1つのプロトタイプとの距離計算を行うコストに等しい．つまり，候補の絞り込みや距離計算の打ち切りによって，距離計算のコストを削減しても，内積計算によってその効果が軽減されてしまう．そこで，本研究では前節の手法を，分割軸を再利用するように改良することで，内積計算の回数を削減する．

図7に示すように，ある分割軸で分割したプロトタイプ集合が，同じ軸に対する分散が大きい場合が存在する．前節の手法では，このような場合にも分割面への射影を行なうため，分割軸の直交補空間から次の分割軸を選択していた．しかし，この場合では新しい軸を選択するより，同じ軸を再利用するほうが効率が良いと考えられる．

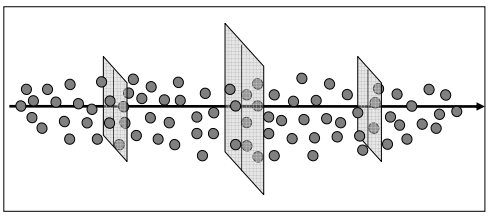


図 7: 分割軸の再利用を行うべき例

そこで，改良手法では主成分分析により得られた固有ベクトルを分割軸と決定した際に，プロトタイプ集合の分割軸成分の標準偏差 (固有値の平方根で与えられる) を記憶し，それを新しい分割軸を選択するかどうかの判定に用いる．

以下に，改良後の分割軸の決定アルゴリズムを延べる．

1. ルートノードからあるノードまでのパス上のノードで選択された分割軸の集合を $D = \{\boldsymbol{d}_1, \cdots, \boldsymbol{d}_n\}$ とし，それぞれに対応する標準偏差を $S = \{s_1, \cdots, s_n\}$ とする．
2. 領域内のプロトタイプ集合に対して主成分分析を行い，最大固有値 λ と対応する第一固有ベクトル e を求める．また固有ベクトルから標準偏差 $\hat{s} = \sqrt{\lambda}$ を計算する．
3. $s_{\max} = \max S \leq W\hat{s}$ なら e をそのノードの分割軸とし，D, S に $\hat{s}$ と ϕ を追加する．

$$\boldsymbol{d} = e, \quad (12)$$

$$D = D \cup \{e\}, \quad (13)$$

$$S = S \cup \{\hat{s}\} \quad (14)$$

$s_{\max} = \max S > W\hat{s}$ なら $s_{\max}$ に対応する分割軸をそのノードの分割軸とし，標準偏差を半分にする．

$$i^* = \arg\min_i s_i, \quad (15)$$

$$\boldsymbol{d} = \boldsymbol{d}_{i^*}, \quad (16)$$

$$s_{i^*} = \frac{s_{i^*}}{2} \quad (17)$$

ただし，$W(0 < W \leq 1)$ は新しい分割軸を選択する度合をコントロールするための重み係数である．

プロトタイプの標準偏差を基準として，分割軸を決定することで，分割軸の再利用により内積計算回数を減らすだけでなく，空間分割の効率を上げる効果も得ることができる．

プロトタイプの分布に大きな偏りがなく，単純に標準偏差を基準として，分割軸を選択するだけでは，分割軸の再利用があまり行われない場合も考えられる．そのような場合には，上記のアルゴリズムの 3 で $\hat{s}$ にかける重み W の値を小さくすることで，分割軸が再利用されやすくすることができる．

4 実験結果

4.1 実験環境

提案手法の有効性を確認するため，次元数，プロトタイプ数を変化ながて実験を行った．対象データとして，顔画像として CAS-PEAL のグレースケール人物画像 [7] の顔部分だけを切り出したものを用い，顔画像のスケーリングによって次元数を変化させた．

以下に実験に用いた画像データの概要を示す．

対象データ グレースケール顔画像 (CAS-PEAL より)

次元数 (画像サイズ)

36 次元 (6×6) から 4096 次元 (64×64)

プロトタイプ数 1000 から 5000

入力パターン数 2000

また，プロトタイプに用いる学習用データと，入力パターンとして用いるテスト用データは異なる顔画像とし，2000 個の入力パターンに対する平均値を結果として示す．

なお，実験には Pentium(R) 4 2.60GHz CPU，1.0 GB memory を用いた．

4.2 分割軸決定時の重みの検討

まず，3.4 節に示す，分割軸を決定する際の重み W の影響について調べた．

図 8 に，次元数 1024(32)，プロトタイプ数 1000 の場合に，重みを変化させながら探索を行ったときの探索時間の変化を示す．この結果より，重みの値を 0.05 より小さくすると ANN よりも高速な探索が可能となっていることがわかる．しかし，さらに重みを小さくして，0.01 よりも小さな値にすると，かえって探索時間が大きくなる場合も見られた．これは，分割軸に用いる基底数が少なすぎて，最近傍候補の絞り込みや距離計算の打ち切りの効果が軽減されたためと考えられる．

この結果をもとに，以降の実験では重み W を 0.01 とした．

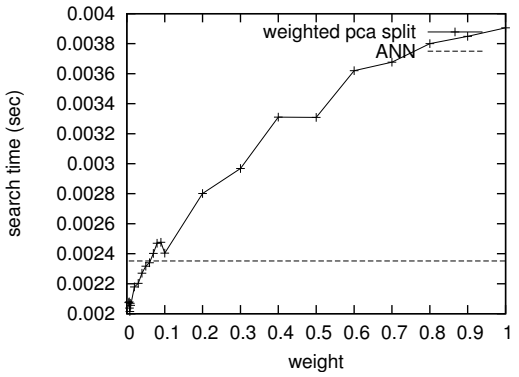
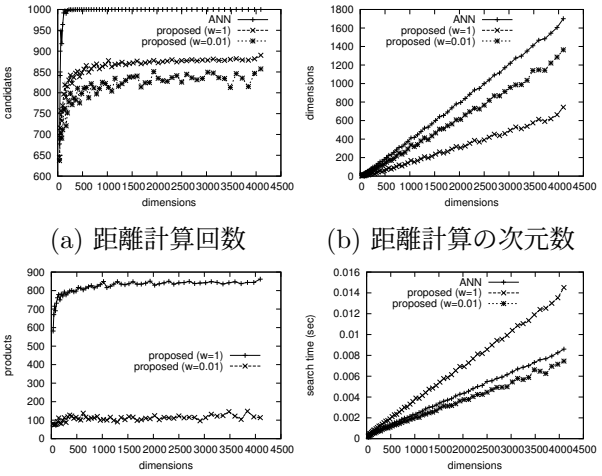


図 8: 重みの変化に対する探索時間の変化



(a) 距離計算回数　(b) 距離計算の次元数

(c) 内積回数　(d) 探索時間

図 9: 次元数を変化させた場合の実行結果

4.3 次元数の変化に対する評価

次にプロトタイプ数を 1000 に固定し，次元数を 32 次元 (6×6) から 4096(64×64) に変化させた場合について，ANN と提案手法の比較実験を行った．

図 9 に比較結果を示す．(a) の距離計算回数を見ると，ANN は 100 次元程度でほぼ全探索となっているのに対し，提案手法は 8 割程度のプロトタイプとの距離計算で済んでいることがわかる．また，距離計算に用いた次元数に関しても提案手法のほうが ANN より少なく，効率的な探索が行われていることがわかる．

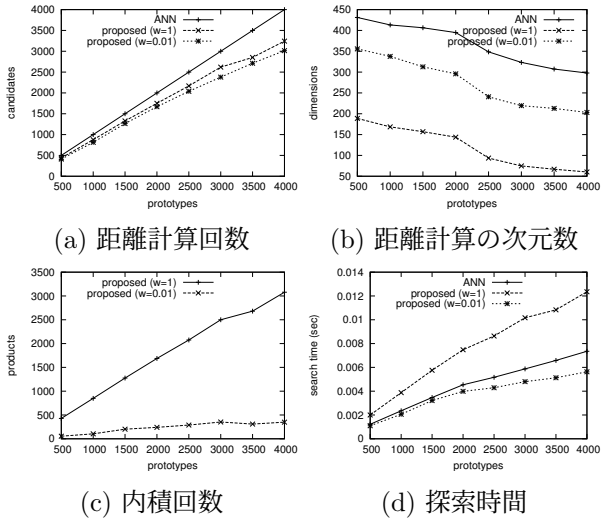
また，距離計算回数と距離計算に用いた次元数だけで見ると，重みなしのほうが効率的であるが，(c) の内積回数を見ると圧倒的に重みなしのほうが内積回数が少ないことがわかる．

(d) 実際の探索時間を示す．この結果より，低次元では ANN のほうこ高速であるが，高次元になると提案手法で重みを $W = 0.01$ とした場合が最も高速である．これらの結果より，提案手法の有効性が確認できる．

4.4 プロトタイプ数の変化に対する評価

次に，次元数を 1024 次元 (32×32) に固定し，プロトタイプ数を 1000 から 5000 まで変化させた場合の結果を図 10 に示す．

この場合でも，距離計算回数，距離計算に用いた次元数ともに，提案手法のほうが ANN より少ないことがわかる．また，探索時間においても，提案手法で $W = 0.01$ とした場合が最も高速であることがわかる．



(a) 距離計算回数　(b) 距離計算の次元数

(c) 内積回数　(d) 探索時間

図 10: プロトタイプ数を変化させた場合の実行結果

5 まとめ

本論文では，高次元空間での最近傍探索において，各領域毎にプロトタイプの分散を最大化し，かつ互いに直交する軸を分割軸として，再帰的に選択する再帰的直交変換によって，ANN が持つ最近傍候補の絞り込みと，距離計算の打ち切りの両方の効果を向上させる手法を提案した．また，分割軸を再利用しながら効率良く分割軸を構成することで分割軸との内積回数を削減する手法について示した．実験では CAS-PEAL の顔画像に対して探索実験を行い，高次元のデータに対して ANN より高速であることを確認した．

今後の課題として，探索時に必要な内積計算のコストの削減が挙げられる．現在分割軸の選択時の重みによって，分割軸の再利用率を上げることで内積計算の回数を削減しているが，やはりまだ全体の処理に対する内積計算のコストが大きく十分に効果的であるとは言えない．この問題を解決するために，高速演算が可能な基底を分割軸として選択することで，内積計算自体のコストを抑える方法について検討する．

謝辞

本研究の一部は，文部科学省科学研究費補助金基盤研究 (A)(2)16200014，及び若手研究 (B)16700143 の補助を受けている．

参考文献

[1] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A.Y. Wu, “An optimal algorithm for approximate nearest neighbor searching,” Journal of the ACM, Vol.45, pp.891-923, 1998

[2] Library for Approximate Nearest Neighbor Searching (http://www.cs.umd.edu/mount/ANN/)

[3] J.L.Bentley: Multidimensional binary search trees used for associative searching, Communications of the ACM, Vol.18, No.9, pp.119-139, 1997

[4] LAESA : Linear Nearest-Neighbor Approximating and Eliminating Search Algorithm Mico, M.L., J. Oncina, and E. Vidal: A new version of the nearest-neighbour approximating and eliminating search algorithm (AESA) with linear preprocessing time and memory requirements. Pattern Recognition Letters (1994)15 9-17.

[5] J.K.Uhlmann, ”Satisfying General Proximity/Similarity Queries with Metric Trees”, Information Processing Letters, Vol40, pages175-179,1991.

[6] S.Brin, ”Near neighbor search in large metric spaces.” In Proc. 21th Conference on Very Large Databases (VLDB 95), pages 574-584,1995.

[7] Wen.Gao, Bo Cao, Shiguang Shan, et.al ”The CAS-PEAL Large-Scale Chinese Face Datebase and Baseline Evaluation” (http://www.jdl.ac.cn/peal/index.html)

[8] 北研二，獅々堀正幹，大恵俊一郎：“ 多次元データの高速近傍検索アルゴリズム ”，情報処理学会研究報告，2003-FI-72, 2003-NL-157, pp.9-16, 2003.

[9] D. I. Barnea and H. F. Silverman. A class of algorithms for fast digital image registration. IEEE Trans. on Comput., C- 21(2):179-186, 1972.