2E-6

kd-treeによる構造化とGPUを用いた 大規模高次元ベクトル群のための高速近傍探索法

松村　聖司† 毛受　崇† 赤間　浩樹† 松尾　嘉典† 奥村　昌和† 山室　雅司†
†日本電信電話株式会社　NTTサイバースペース研究所

1. はじめに

局所特徴量を用いた物体認識処理は，局所特徴量抽出と照合処理の 2 つの処理に大別される．この内，照合処理中の最近傍探索においては，照合対象となる DB（データベース）中の画像群が増大した際，DB ベクトル数も増える為，処理コストが増大するという問題がある．従来手法では探索範囲を大きく削減し，探索精度を犠牲にして，探索速度を高速化している為，物体認識の精度に悪影響を及ぼす可能性がある．そこで本研究では，大規模な高次元ベクトル群に対して，探索精度の低下を抑えた高速な最近傍探索手法について検討する．

2. 提案手法

提案手法を実現する為に 2 つの点に着目した．従来の最近傍探索法[1]である kd-tree[2] based ANN（Apporoximate Nearest Neighbor）の探索範囲を絞る点と， GPU（Graphic Processor Unit）を用いることでユークリッド距離計算を並列処理する点である．これら 2 つの着目点から，探索範囲は大きめに絞り，絞った範囲を GPU による並列処理を用いて高速に探索する手法を提案する．探索範囲を広くとることで，距離計算できる DB ベクトル数が増え，結果として探索精度も高く保つことができる．本提案手法を，GPU を用いた kd-tree 構造化データ高速探索手法（図 1）と呼ぶこととする．

本提案手法は，kd-tree 作成と最近傍探索の 2 段階に分けることができる．まず kd-tree 作成時について，従来手法では，葉ノードに対応するベクトルが 1 つになるまで，ベクトル空間を分割する．一方，本手法では，ベクトル空間の分割を途中で止めることで，葉ノードに複数のベクトルを対応させる．

次に最近傍探索時は，各クエリベクトルについて kd-tree を探索し，到達した葉ノード内の DB ベクトル群と GPU を用いて距離計算し，最短距離値 r とそれに対応した DB ベクトルを求める．そして，最短距離値 r とバックトラック半径度合 α を用いて，バックトラック半径 r’（= α・r）でバックトラック処理を行う．バックトラック対象となった葉ノード群内の DB ベクトル群に対して GPU で並列距離計算を行い，その時の最短距離値 r’’とそれに対応した DB ベクトルを求める．最後に，最初に到達した葉ノード内の最短距離値 r とバックトラック対象の葉ノード群内の最短距離値 r’’の大小を比較し，値の小さい方に対応する DB ベクトルを最近傍ベクトルとして出力する．

3. 設計

提案手法を実現する為のシステムを設計する（図 2）．システム全体は集計ノード 1 台と複数基の GPU を搭載した分散探索ノード複数台から構成される．集計ノードは，クエリベクトル群の入力と物体認識結果の出力を行う．

高速な探索処理を実現する為には，GPU のグローバルメモリ上に DB ベクトル群を保持させておく必要がある．また，今後 DB ベクトル数が増大することが想定されるので， GPU を搭載したマシンを複数台用意し，DB ベクトル群を水平分割し保持させる必要がある．この為，探索ノードを複数台用意する．これによって，大規模なベクトル群に対する最近傍探索の GPU による高速な並列化探索が可能になる．各分散探索ノードは，各ノードが保持する分割 DB ベクトル群に対して最近傍探索を行う．

4. 評価実験

4.1. 実験

本提案手法ではパラメータとして葉ノードサイズとバックトラック半径度合 α の 2 つが存在する。これら 2 つのパラメータを変化させた時に探索時間と探索精度がどのように変化するかを測定する．そして，精度固定でみた場合に，従来手法の kd-tree based ANN に対して，本提案手法がどの程度高速になっているかを検証する．

実験環境としては，DB 側で 1 億 240 万のベクトル群をシステム全体で取り扱う．このベクト

Method of Nearest Neighbor Search for High Dimensional Vectors using kd-tree Structure and GPU
Seiji MATSUMURA†, Takashi MENJO†, Hiroki AKAMA†, Yoshinori MATSUO†, Masakazu OKUMURA†, Masashi YAMAMURO†
†NTT Cyber Space Laboratories

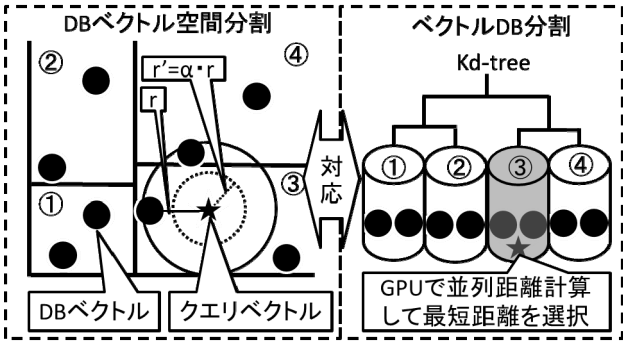

図 1．GPU を用いた
kd-tree 構造化データ高速探索手法

ル群を，表 1 のスペックの分散探索ノード 10 台に水平分割し，分散探索ノード 1 台あたり，1024 万ベクトルずつ保持させる．GPU デバイスは 2 基使用し，1 基あたり，512 万 DB ベクトルをグローバルメモリ上に保持させて，GPU デバイス単位で kd-tree を作成する．

4.2. 結果

測定結果を図 3 に示す．精度一定でみたときに，高精度領域において，従来手法より高速であることを確認した．本実験においては，精度約 40 % 以上で，従来手法よりも高速であった．例えば，探索時間 1000 [ms] でみた時の提案手法の精度約 60 % においては，約 15 % の高速化が確認できた．

4.3. 考察

本実験によって，高精度領域において，従来手法よりも高速であることが確認できた．しかし，本手法は，さらに高速化できる可能性がある．それは，バックトラックによって生じる CPU－GPU 間の転送コストを，転送処理と他の処理とを並行に実行することで，処理時間に含めない方法である．例えば，葉ノードの DB ベクトル数が 1250 個の場合に，転送処理と他の処理を並行に処理できれば，約 70 [ms] 短縮できる可能性がある（図 4）．

5. まとめと今後の課題

本研究では，GPU を用いた kd-tree 構造化データの高速探索手法を提案した．実験の結果，高精度領域において，従来手法より高速であることを確認した．

今後の課題として，本研究の実装では，DB ベクトル数が増加すると，分散探索ノードも増加することになる．その為，集計ノードと分散探索ノード間の通信がボトルネックとなることが懸念される．この問題を評価する為にも，DB ベクトル数を今以上に増やした場合の評価が必要である．

表 1．実験時の分散探索ノードのスペック

	ホスト	GPUデバイス
processor	Intel® Xeon® X5650 6 Cores @ 2.67GHz	NVIDIA Tesla M2090 512 CUDA Cores @ 1.3 GHz
memory	24GB	5.6GB (グローバルメモリ)

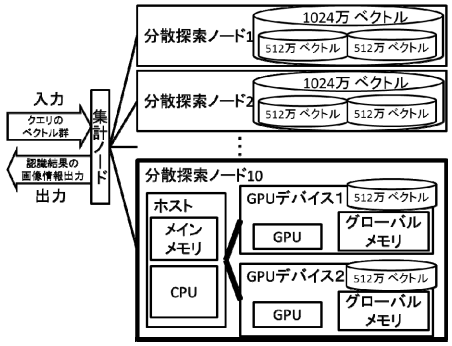

図 2．全体構成と分散探索ノードの構成

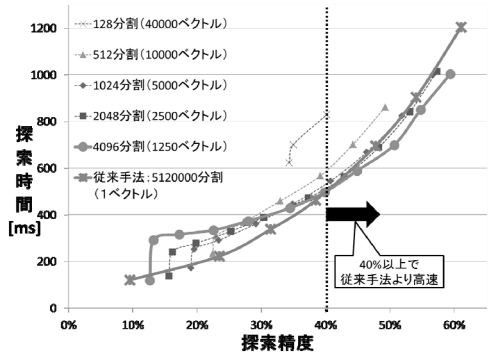

図 3．各探索精度における探索速度の比較

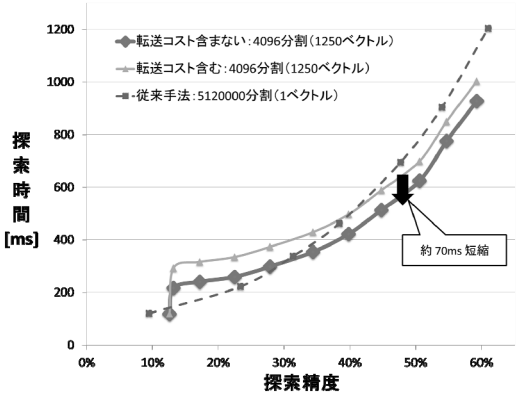

図 4．転送コストを含めない場合の推定処理時間

参考文献

[1] 和田俊和，“最近傍探索の理論とアルゴリズム”，2009 情報処理学会 CVIM 研究会，vol.2009-CVIM-169, no.12, pp.1-12, Nov.2009.

[2] P. Indyk, R. Motwani, P. Raghavan, and S. Vempala, Locality-preserving hashing in multidimentional spaces, In Proceedings of 29th ACM Symposium on Theory of Computing, pp.618-625, 1997.