(12) 特許協力条約に基づいて公開された国際出願

(19) 世界知的所有権機関
国際事務局

(43) 国際公開日
2013 年 9 月 6 日(06.09.2013)

WIPO | PCT

(10) 国際公開番号
WO 2013/129580 A1

(51) 国際特許分類:
G06F 17/30 (2006.01)

(21) 国際出願番号: PCT/JP2013/055440

(22) 国際出願日: 2013 年 2 月 28 日(28.02.2013)

(25) 国際出願の言語: 日本語

(26) 国際公開の言語: 日本語

(30) 優先権データ:
特願 2012-042172 2012 年 2 月 28 日(28.02.2012) JP
61/684,911 2012 年 8 月 20 日(20.08.2012) US

(71) 出願人: 公立大学法人大阪府立大学(OSAKA PREFECTURE UNIVERSITY PUBLIC CORPORATION) [JP/JP]; 〒5998531 大阪府堺市中区学園町１番１号 Osaka (JP).

(72) 発明者: 岩村 雅一(IWAMURA, Masakazu); 〒5998531 大阪府堺市中区学園町１番１号 公立大学法人大阪府立大学内 Osaka (JP). 佐藤 智一(SATO, Tomokazu); 〒5998531 大阪府堺市中区学園町１番１号 公立大学法人大阪府立大学内 Osaka (JP). 黄瀬 浩一(KISE, Koichi); 〒5998531 大阪府堺市中区学園町１番１号 公立大学法人大阪府立大学内 Osaka (JP).

(74) 代理人: 野河 信太郎, 外(NOGAWA, Shintaro et al.); 〒5300047 大阪府大阪市北区西天満５丁目１６－３ 西天満ファイブビル 野河特許事務所 Osaka (JP).

(81) 指定国 (表示のない限り、全ての種類の国内保護が可能): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

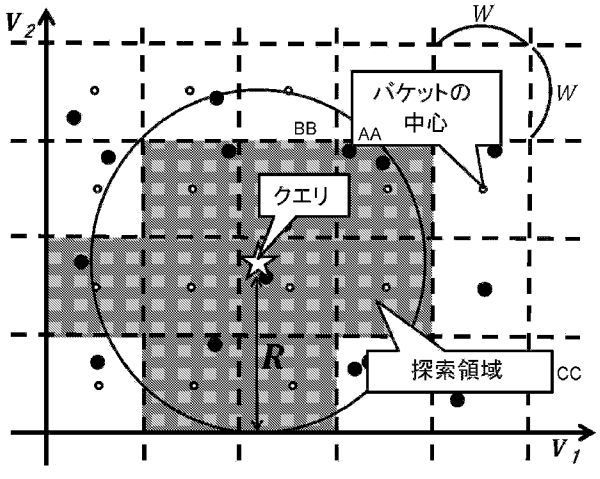
(84) 指定国 (表示のない限り、全ての種類の広域保護が可能): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), ユーラシア (AM, AZ, BY, KG, KZ, RU, TJ, TM), ヨーロッパ (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

[続葉有]

(54) Title: APPROXIMATE NEAREST NEIGHBOR SEARCH DEVICE, APPROXIMATE NEAREST NEIGHBOR SEARCH METHOD, AND PROGRAM

(54) 発明の名称 : 近似最近傍探索装置、近似最近傍探索方法およびそのプログラム

[図3]



AA... BUCKET CENTER
BB... QUERY
CC... SEARCH REGION

(57) Abstract: An objective of the present invention is to implement an approximate nearest neighbor search rapidly and with high precision in searching by appropriately reducing the number of nearest neighbor candidates. An approximate nearest neighbor search device is applied which comprises: a database storage unit which, when a plurality of points which are represented with vector data is inputted, computes a hash index by applying a hash function to each point, and stores each point in a multi-dimensional hash table by projecting each point in a multi-dimensional space which is segmented into a plurality of regions by the multi-dimensional hash table bins; a search range establishment unit which, when a query is inputted, applies the hash function to the query, establishes a location of the query within the space, establishes estimate values of the distance from the query to each region within the space, and establishes regions to be searched on the basis of the estimate values; and a nearest neighbor establishment unit which calculates the distance from each point within the search region to the query, and computes the nearest point to the query to be the nearest neighbor to the query. The search range establishment unit refers to the index of each region and derives a representative point of the region, establishes the estimate value on the basis of the distance between the query and each representative point, applies a branch and bound technique, excluding the regions which cannot be the regions to be searched, and establishes the regions to be searched.

(57) 要約:

[続葉有]

WO 2013/129580 A1

添付公開書類:

— 国際調査報告（条約第 21 条(3)）

最近傍候補の数を適切に絞り込むことにより、高い探索精度と高速性を兼ね備えた近似最近傍探索を実現する。 ベクトルデータで表現される複数の点が入力されたとき、各点にハッシュ関数をそれぞれ適用してハッシュのインデックスを算出し、前記多次元ハッシュテーブルのビンによって複数の領域に分割された多次元空間内に各点を射影することにより各点を多次元ハッシュテーブルに格納してなるデータベース格納部と、クエリが入力されたとき、そのクエリに前記ハッシュ関数を適用して前記空間内でのクエリの位置を決定し、クエリと前記空間内の各領域との距離の推定値を決定し、その推定値に基づいて探索すべき領域を決定する探索範囲決定部と、前記探索領域内の各点とクエリとの距離を計算し、クエリに最も近い点をクエリの最近傍点として算出する最近傍点決定部とを備え、前記探索範囲決定部は、各領域のインデックスを参照してその領域の代表点を求め、前記クエリと各代表点との距離に基づいて前記推定値を決定し、分枝限定法を適用して前記探索すべき領域になり得ない領域を除外して前記探索すべき領域を決定する近似最近傍探索装置を適用する。

明　細　書

発明の名称 ：

近似最近傍探索装置、近似最近傍探索方法およびそのプログラム

技術分野

[0001] この発明は、近似最近傍探索装置、近似最近傍探索方法およびそのプログラムに関し、より詳細にはコードブック付きハッシングによる高速距離推定を用いた近似最近傍探索装置、近似最近傍探索方法および近似最近傍探索プログラムに関する。

背景技術

[0002] 近年の情報処理において大規模データの処理は不可欠であり、年々その重要度が増している。この要因のひとつはハードウェアである。すなわち、計算機環境の高性能化、低価格化、記憶媒体の大容量化によって大容量のデータを扱えるようになり、さらに言えば現実的な時間で処理できるようになったことである。他の要因と言えるのはコンテンツとニーズである。すなわち、一般の人々が写真や動画、音楽などの様々なコンテンツを自ら制作し、Flickr（URL: http://www.flickr.com/ 参照）やYouTube（URL: http://www.youtube.com/ 参照）などのサイトにアップロードするようになったことに加えて、それらの中から自分の興味に合致するものを高速で発見したいという欲求が生じたことである。このような写真や動画に限らず、人類が扱うことができるデータの総量は猛烈な勢いで増え続けているため、大規模データの処理技術の開発は喫緊の課題である。

[0003] この課題の一つの答えになるのが最近傍探索である。最近傍探索はベクトルで表現されるデータ（点）の中から、クエリと最も類似している、即ち最も距離が小さいデータ（最近傍点）を探す手法である。最近傍探索は、大規模データの処理を実現する基本的で有効な技術であり、基本的であるが故に優れた手法には幅広い応用可能性がある。発明者らが携わっているものだけでも応用先として物体認識（例えば、非特許文献１参照）、文書画像検索（

例えば、非特許文献２参照）、文字認識（例えば、非特許文献３参照）および顔認識（例えば、非特許文献４参照）を挙げることができる。いずれも比較的大規模なデータに対して非常に高速に動作することが示されている。前述の技術は、広い意味での画像認識技術といえるが、最近傍探索の応用分野はそれに留まらず、統計分類、符号論理、データ圧縮、レコメンデーションシステム、スペルチェッカ等におよぶ。

実応用を考えると、最近傍探索には高速性が求められる。高速化が進めば、同じ時間でこれまで以上に大規模なデータを処理することができ、従来は処理時間の制約から導入を諦めていた用途にも使用できるようになる。

[0004] この要求を満たすために生み出されたのが近似最近傍探索である。近似最近傍探索とは、最近傍探索に近似を導入し、探索誤りを許容することで処理時間を大幅に削減することができる技術である。探索誤りが生じるため、必ずしも真の最近傍点がみつかる保証はないが、ある程度の探索精度が得られればよいアプリケーションに使用される。探索精度の向上と高速化は相反する要求であり、一般に両者を同時に満足することは容易でない。

[0005] これまでに提案された代表的な近似最近傍探索手法について述べる。木構造を用いた手法として、Approximate Nearest Neighbor（あるいはＡＮＮ、例えば、非特許文献５参照）およびFast Library for Approximate Nearest Neighbors（あるいはＦＬＡＮＮ、例えば、非特許文献６参照）などが知られている。また、ハッシュ法を用いた手法としてLocality Sensitive Hashing（あるいはＬＳＨ例えば、非特許文献７、８および９参照）、Spectral Hashing（あるいはＳＨ、例えば、非特許文献１０参照）ならびに佐藤らの手法（例えば、非特許文献１１参照）などが知られている。

さらに、ＦＬＡＮＮに採用されているパラメータチューニングの手法としてrandamized kd-tree（例えば、非特許文献１２参照）や階層的k-means（例えば、非特許文献１３参照）が知られている。

先行技術文献

非特許文献

[0006] 非特許文献1：黄瀬浩一、野口和人、岩村雅一、”参照特徴ベクトルの増加による低品質画像の高速・高精度認識、”電子情報通信学会論文誌D、vol.J93-D, no.8, pp.1353-1363, Aug. 2010.

非特許文献2：K. Takeda, K. Kise, and M. Iwamura, ”Real-time document image retrieval for a 10 million pages database with a memory efficient and stability improved llah,” Proc. International Conference on Document Analysis and Recognition (ICDAR2011) , pp.1054-1058, Sept. 2011.

非特許文献3：M. Iwamura, T. Tsuji, and K. Kise, ”Memory-based recognition of camera-captured characters,” Proceedings of the 9th IAPR International Workshop on Document Analysis Systems (DAS2010) , pp.89-96, June 2010.

非特許文献4：前川敬介、内海ゆづ子、岩村雅一、黄瀬浩一、”100万顔画像データベースに対する34msでの照合の実現～近似最近傍探索を用いた大規模高速顔画像検索～、”電子情報通信学会技術研究報告、vol.111, no.353, pp.95-100, Dec. 2011.

非特許文献5：S. Arya, D.M. Mount, N.S. Netanyahu, R. Silverman, and A.Y. Wu, ”An optimal algorithm for approximate nearest neighbor searching in fixed dimensions,” Journal of the ACM, vol.45, no.6, pp.891-923, Nov. 1998.

非特許文献6：M. Muja and D.G. Lowe, ”Fast approximate nearest neighbors with automatic algorithm configuration,” International Conference on Computer Vision Theory and Application (VISSAPP'09), pp.331-340, INSTICC Press, 2009.

非特許文献7：P. Indyk and R. Motwani, ”Approximate nearest neighbors: towards removing the curse of dimensionality,” Proceedings of the thirtieth annual ACM symposium on Theory of computing, pp.604-613 1998.

非特許文献8：M. Datar, N. Immorlica, P. Indyk, and V.S. Mirrokni, ”Lo

cality-sensitive hashing scheme based on p-stable distributions," Proceedings of the twentieth annual symposium on Computational geometry, pp.253-262 2004.

非特許文献9：A. Andoni and P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," Annual IEEE Symposium on Foundations of Computer Science, pp.459-468 2006.

非特許文献10：Y. Weiss, A. Torralba, and R. Fergus, "Spectral Hashing," Advances in Neural Information Processing Systems, pp.1753-1760, 2008.

非特許文献11：佐藤智一、武藤大志、岩村雅一、黄瀬浩一、"バケット距離に基づく近似最近傍探索、"第3回データ工学と情報マネジメントに関するフォーラム論文集、pp.1-6, Feb. 2011.

非特許文献12：C. Silpa-Anan and R. Hartley, "Optimised kd-trees for fast image descriptor matching," IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2008), pp.1-8 2008.

非特許文献13：B. Leibe, K. Mikolajczyk, and B. Schiele, "Efficient clustering and matching for object class recognition," Proc. BMVC, pp.1-10, 2006.

発明の概要

発明が解決しようとする課題

[0007] 近似最近傍探索では予めデータ空間を多数の領域に分割してインデクシングを施す。この発明は、特にハッシュを使用する手法に係るものである。ハッシュを用いた近似最近傍探索は、基本的にデータ空間を複数の軸に沿って線形分割し、分割された領域（ビン）ごとにデータをハッシュテーブルに登録する。そして、クエリの入った領域または、その近傍の領域に入った点をハッシュテーブルから抽出し、その中から最近傍点を求めることで、高速な探索を実現している。複数のハッシュテーブルを用いる場合はビンの積領域（バケット）が分割の単位になる。

一般に近似最近傍探索は二段階の処理から成り、一段目で最近傍点が含まれる確率の高いビンまたはバケットを特定する。そこに属する点を最近傍候補と呼ぶことにする。二段目では最近傍候補に含まれる点のうち、最もクエリからの距離が小さい点を算出する。つまり、「近似」の要素が入るのは一段目の処理のみである。従って、この処理が近似最近傍探索の探索精度と処理時間を左右する。

[0008] 一段目の最近傍候補の特定では距離計算の対象を最近傍候補に限定することで距離計算に要する処理時間を大幅に削減する。しかし、これは諸刃の剣であり、最近傍候補の数を減らすことで高速化を図れるものの、探索精度が低下してしまう。それは最近傍候補から真の最近傍点が漏れてしまい、最近傍探索に失敗する確率が上昇するためである。従って、一段目の処理において求められるのは最近傍候補の数を絞りつつも真の最近傍点を漏らさないことである。しかも最近傍候補の選定は高速に行う必要がある。

[0009] これらの要求を満足するために発明者らが採った方針は、最近傍候補の選定の際に二段目で使用するのと同じ距離尺度を用いることである。これにより、最近傍候補の数を絞っても真の最近傍点が最近傍候補に含まれる確率は減少し難い。しかし、このような処理には通常大きな計算コストが必要になる。この課題を解決するためには、前述の処理を高速に実行できる近似最近傍探索の手法が望まれている。

この発明は、以上のような事情を考慮してなされたものであって、最近傍候補を適切に絞り込むことにより、高い探索精度と高速性を兼ね備えた新たな近似最近傍探索の手法を提供するものである。

課題を解決するための手段

[0010] ベクトルデータで表現される複数の点が入力されたとき、多次元ハッシュテーブルのインデックス算出用のハッシュ関数をそれぞれ適用して各点についてハッシュインデックスを算出し、前記多次元ハッシュテーブルのビンによって複数の領域に分割された多次元空間内の前記ハッシュインデックスに応じた領域に各点を射影することにより各点を多次元ハッシュテーブルに格

納してなるデータベース格納部と、クエリが入力されたとき、そのクエリに前記ハッシュ関数を適用して前記多次元空間内でのクエリの位置を決定し、クエリと前記空間内の各領域との距離の推定値を決定し、その推定値に基づいて少なくとも１つの探索すべき領域を決定する探索範囲決定部と、前記探索すべき領域内の各点とクエリとの距離を計算し、クエリに最も近い点をクエリの最近傍点として算出する最近傍点決定部とを備え、前記探索範囲決定部は、各領域のインデックスを参照してその領域の代表点を求め、前記クエリと各代表点との距離に基づいて前記推定値を決定し、分枝限定法を適用して前記探索すべき領域になり得ない領域を除外して前記探索すべき領域を決定することを特徴とする近似最近傍探索装置を提供する。

言い換えればこの発明は、ベクトルデータで表現される複数の点が入力されたとき、各点にハッシュ関数をそれぞれ適用してハッシュのインデックスを算出し、正規直交基底によって複数の領域に分割された多次元空間内の前記インデックスに応じた領域に各点を射影することにより各点が多次元ハッシュテーブルに登録されてなるデータベース格納部と、クエリが入力されたとき、そのクエリに前記ハッシュ関数を適用して前記空間内でのクエリの位置を決定し、クエリと前記空間内の各領域との距離の推定値を決定し、その推定値に基づいて１以上の領域を探索領域として決定する探索範囲決定部と、前記探索領域内の各点とクエリとの距離を計算し、クエリに最も近い点をクエリの最近傍点として算出する最近傍点決定部とを備え、前記探索範囲決定部は、各点が属する領域のインデックスを参照してその領域を代表する代表点の位置を求め、前記推定値を決定することを特徴とする近似最近傍探索装置を提供する。

[0011] また、異なる観点から、この発明は、コンピュータが、ベクトルデータで表現される複数の点が入力されたとき、多次元ハッシュテーブルのインデックス算出用のハッシュ関数をそれぞれ適用して各点についてハッシュインデックスを算出し、前記多次元ハッシュテーブルのビンによって複数の領域に分割された多次元空間内の前記ハッシュインデックスに応じた領域に各点を

射影することにより各点を多次元ハッシュテーブルに格納してなるデータベース格納部にアクセスするステップと、クエリが入力されたとき、そのクエリに前記ハッシュ関数を適用して前記空間内でのクエリの位置を決定し、クエリと前記空間内の各領域との距離の推定値を決定し、その推定値に基づいて少なくとも１つの探索すべき領域を決定する探索範囲決定ステップと、前記探索すべき領域内の各点とクエリとの距離を計算し、クエリに最も近い点をクエリの最近傍点として算出するステップとを備え、前記探索範囲決定ステップは、各領域のインデックスを参照してその領域の代表点を求め、前記クエリと各代表点との距離に基づいて前記推定値を決定し、分枝限定法を適用して前記探索すべき領域になり得ない領域を除外して前記探索すべき領域を決定することを特徴とする近似最近傍探索方法を提供する。

言い換えれば、コンピュータが、ベクトルデータで表現される複数の点が入力されたとき、各点にハッシュ関数をそれぞれ適用してハッシュのインデックスを算出し、正規直交基底によって複数の領域に分割された多次元空間内の前記インデックスに応じた領域に各点を射影することにより各点が多次元ハッシュテーブルに登録されてなるデータベース格納部にアクセスするステップと、クエリが入力されたとき、そのクエリに前記ハッシュ関数を適用して前記空間内でのクエリの位置を決定し、クエリと前記空間内の各領域との距離の推定値を決定し、その推定値に基づいて１以上の領域を探索領域として決定する探索範囲決定ステップと、前記探索領域内の各点とクエリとの距離を計算し、クエリに最も近い点をクエリの最近傍点として算出するステップとを備え、前記探索範囲決定ステップは、各領域のインデックスを参照してその領域を代表する代表点の位置を求め、前記クエリの位置と各代表点の位置との差を前記推定値とすることを特徴とする近似最近傍探索方法を提供する。

[0012] さらに異なる観点から、この発明は、ベクトルデータで表現される複数の点が入力されたとき、多次元ハッシュテーブルのインデックス算出用のハッシュ関数をそれぞれ適用して各点についてハッシュインデックスを算出し、

前記多次元ハッシュテーブルのビンによって複数の領域に分割された多次元空間内のハッシュインデックスに応じた領域に各点を射影することにより各点を多次元ハッシュテーブルに格納してなるデータベース格納部にアクセスする処理と、クエリが入力されたとき、そのクエリに前記ハッシュ関数を適用して前記空間内でのクエリの位置を決定し、クエリと前記空間内の各領域との距離の推定値を決定し、その推定値に基づいて少なくとも１つの探索すべき領域を決定する探索範囲決定部としての処理と、前記探索すべき領域内の各点とクエリとの距離を計算し、クエリに最も近い点をクエリの最近傍点として算出する最近傍点決定部としての処理をコンピュータに実行させ、前記探索範囲決定部は、各領域のインデックスを参照してその領域の代表点を求め、前記クエリと各代表点との距離に基づいて前記推定値を決定し、分枝限定法を適用して前記探索すべき領域になり得ない領域を除外して前記探索すべき領域を決定することを特徴とする近似最近傍探索プログラムを提供する。あるいは、そのプログラム製品を提供する。

上述の３つの観点からの発明は、いずれも後述の実施形態における「第１の改良点：ハッシュに基づく点対バケットの距離推定」の記載に関連する。

発明の効果

[0013] この発明の近似最近傍探索装置において、探索範囲決定部は、各領域のインデックスを参照してその領域の代表点を求め、前記クエリと各代表点との距離に基づいて前記推定値を決定するので、クエリと各領域との距離計算を行わずとも、インデックスを用いて前記推定値を決定し、その推定値に基づいて探索すべき領域（探索領域）を決定し、最近傍点決定部が距離計算を行うべき点を絞り込むことができる。また、バケット対バケットの距離推定を行う佐藤らの手法に比べて、点対バケットの距離を推定するこの発明の手法は同じデータ構造のまま距離推定の精度を向上させることができる。さらに、分枝限定法を適用して探索領域になり得ない領域を除外するので、短時間で探索領域を決定できる。

[0014] この発明において、データベースに登録される個々のデータおよびデータ

ベースの検索に用いられるクエリ（検索質問）のデータは、少なくとも一つの点として表現される。各点は、前記データあるいはクエリの特徴を示す属性を有しており、その属性はベクトルデータで表現される。ハッシュは、メモリ上でデータを高速に検索するための公知の手法である。ハッシュ関数は、ベクトルデータを入力としてスカラー値を出力する。そのスカラー値はデータテーブルの一種であるハッシュテーブルを参照するために用いる離散値であり、ハッシュ値、ハッシュインデックスあるいは単にインデックスと呼ぶ。ハッシュ関数は出力の空間をインデックスがとり得る離散値の分だけ分割するものといえる。ハッシュ関数はまた、入力としてのベクトルデータ（点）を出力の多次元空間に射影するものといえる。この発明において、各ベクトルデータは多次元空間に射影される。一つのハッシュ関数の出力は一つのスカラー値であるから、ベクトルデータを例えばν次元（νは２以上の整数）のベクトル空間に射影するためにν個のハッシュ関数を用いる。各点は、ν個のハッシュ関数を適用することによって、ν個の基底で構成され各基底に沿って複数のビンに分割されかつ各ハッシュ関数のインデックスによって何れかのビンが特定されるν次元空間に射影される。各ビンへの各点の登録がハッシュテーブルを用いて表されるので、ハッシュテーブルはν個（ν次元）ある。ただし、いくつかの次元をまとめて一つのハッシュ関数を用いる場合もあり得る。この場合、ハッシュテーブルはν個よりも少ない。

この発明による近似最近傍探索の手法は、物体認識、文書画像検索、文字認識、顔認識、統計分類、符号論理、データ圧縮、レコメンデーションシステム、スペルチェッカ等に適用できる。

[0015] この明細書では一つのハッシュ関数によって分割される各領域をビンと呼ぶ。また、複数のハッシュ関数によってそれぞれ生成されたビンの積領域、即ち複数のハッシュ関数により分割された各領域をバケットと呼ぶ。

この発明に係る近似最近傍探索は、コンピュータが記憶装置等のハードウェア資源を用いつつデータを処理することによって実現される。

図面の簡単な説明

[0016] [図1]従来の近似最近傍探索の一手法である佐藤らの手法において、距離の推定を示す説明図である。

[図2]この発明において、点対バケットの距離推定の一例を示す説明図である。

[図3]この発明において、点対バケットの距離推定の異なる例を示す説明図である。

[図4]従来の佐藤らの手法において、多次元ハッシュの次元数νを変化させたときの、精度と処理時間の関係を示すグラフである。

[図5]この発明において、最適なパラメータにおける精度と処理時間の関係を、従来手法と比較して示す第１のグラフである。

[図6]この発明において、最適なパラメータにおける精度と処理時間の関係を、従来手法と比較して示す第２のグラフである。

[図7]この発明において、距離推定に適したデータ空間の分割の例を示す説明図である。

[図8]この発明において、クエリ周辺の密度に応じて探索半径を変化させる様子を示す説明図である。

[図9]この発明のＢＤＨおよびk-means ＢＤＨによる距離推定の精度を、横軸を符号長、縦軸を相関係数として従来手法のＳＨと比較して示すグラフである。

[図10]この発明において、符号長を120bitとしたときの推定距離と真の距離との関係を示すグラフである。

[図11]この発明のＢＤＨおよびk-means ＢＤＨにおいて、符号長を変化させたときのハッシュ上の距離における最近傍点の平均順位を従来手法のＳＨと比較して示すグラフである。

[図12]この発明において、探索半径Ｒを固定した場合のＢＤＨ、k-means ＢＤＨおよびk-means ＢＤＨ Ｐのそれぞれについて探索精度と処理時間との関係を示すグラフである。

[図13]この発明において、k-means ＢＤＨ ＰＣの探索精度と処理時間との関

係を探索半径Ｒを固定する従来の方法と比較して示すグラフである。

[図14]この発明において、k-means ＢＤＨ ＰＣの探索精度と処理時間との関係を従来手法のＡＮＮおよびＳＨと比較して示すグラフである。

[図15]この発明において、最近傍候補数に関するパラメータｃと処理時間との関係を示すグラフである。

[図16]この発明において、最近傍候補数に関するパラメータｃと精度との関係を示すグラフである。

[図17]この発明において、最近傍候補数に関するパラメータｃと最近傍候補数との関係を示すグラフである。

[図18]この発明において、探索半径Ｒ以内のバケットを高速に特定する２つのアルゴリズムを示す図である。

[図19]この発明において、探索半径を適応的に変化させる２つのアルゴリズムを示す図である。

[図20]この発明の近似最近傍探索装置の応用例としての画像認識装置示すブロック図である。

[図21]従来の近似最近傍探索の一手法であるＬＳＨの説明図である。

[図22]従来の近似最近傍探索の一手法であるＳＨの説明図である。

[図23]従来の近似最近傍探索の一手法である佐藤らの手法において、距離の概算値を示す説明図である。

[図24]この発明において、データ空間中の点の分布の一例を示す説明図である。

[図25]この発明において、処理時間と精度の比較実験の結果を示すグラフである。（SIFT、100万点）

[図26]この発明において、処理時間と精度の比較実験の結果を示すグラフである。（SIFT、1000万点）

[図27]この発明において、処理時間と精度の比較実験の結果を示すグラフである。（SIFT、1億点）

[図28]この発明において、処理時間と精度の比較実験の結果を示すグラフで

ある。（GIST、10万点）

[図29]この発明において、処理時間と精度の比較実験の結果を示すグラフである。（GIST、100万点）

[図30]この発明において、処理時間と精度の比較実験の結果を示すグラフである。（GIST、1000万点）

発明を実施するための形態

[0017] この発明について詳述する前に、この発明の好ましい態様について説明する。

前記データベース格納部は、Ｍ組（Ｍは２以上の自然数）のハッシュ関数群を用いて各ハッシュ関数群に対応するＭ組の多次元ハッシュテーブルのそれぞれに各点が登録されてなり、各ハッシュ関数群は複数のハッシュ関数を組み合わせたものであってもよい。このようにすれば、例えばν個のハッシュ関数（ν＞Ｍ）を分割することなく１つのハッシュ関数群とする場合、各ハッシュテーブルのビンの数をｓとすると、ｓ個のビンに対応するデータ格納領域をν組用意する必要があるため、ハッシュテーブルのサイズのオーダーはＯ（ｓν）となるところ、ハッシュ関数をＭ組に分割することによってそのオーダーをＯ（ｓν/Ｍ）に抑えることができる。即ち、ハッシュテーブルを分割することによって、近似最近傍探索の処理に要する記憶容量を節約することができる。なお、Ｏ（ｓν）は、問題を解くために必要なおおよその計算量の表記方法であって、ｓが定まったときの計算量がｓのν乗のオーダー、即ち、$as^{\nu}+bs^{(\nu-1)}+\cdots+ls^{2}+ms+n$以下で収まることを表す。ここで、a，b，…，l，m，nは定数である。

この態様は、後述する実施形態における「第２の改良点： ハッシュテーブルの分割」の記載に関連する。

[0018] さらに、Ｍ組のハッシュテーブルは、それぞれのハッシュの次元数が略等しくなるように定められてもよい。このときの基底の選択方法は、基底の分散の和が略等しくなるように定められてもよい。具体的には、分割されたＭ組のハッシュテーブルのそれぞれにおいて各基底方向のデータの分散の合計

し、それらの分散の和がなるべく等しくなるように分割されたハッシュテーブルの各組への基底の割り当てを決定する。一般に分散が大きい基底方向の距離は大きくなり易く、分散が小さい基底方向の距離は小さくなり易い傾向がある。そのため、各ハッシュテーブルの分散の和を合計することはハッシュテーブル毎に計算される距離の大きさを揃える効果がある。本明細書の実施例のように距離計算対象の選択や距離計算の打ち切り（距離計算対象の点が探索範囲内に存在しない場合は距離を一定値で置き換える処理）に用いる探索半径Ｒを各ハッシュテーブルで共通にした場合は、各ハッシュテーブルの分散の和を均一化することによって、Ｍ組に分割された各ハッシュテーブルは同じくらい最近傍候補の決定に寄与すると考えられるため、ハッシュテーブル毎にＲに代わるパラメータを設定する必要がなく簡便に計算することができる。

この態様は、後述する実施形態における「第２の改良点： ハッシュテーブルの分割」の記載に関連する。

[0019] また、前記多次元ハッシュテーブルは、各次元に対応するハッシュテーブルを組み合わせてなり、各ハッシュテーブルは各次元に対応する基底をビンで分割し、各ビンの幅は、すべてのビンが等幅とした場合に各ビンに登録されることになる点と、各ビンを代表する代表点との位置の誤差を各ビンについて求め、それらの誤差の和がより小さくなるように各ビンの幅が調整されてもよい。このようにすれば、等分割のビン幅に比べて各点の距離の推定値の真の距離に対する誤差を小さくし、より良い精度で探索領域を決定することができる。各ビンに登録されたベクトルデータの数に多少があると探索領域のバケットに登録されたデータ数の多少によって探索領域内の各点との距離計算の処理時間にバラツキが生じるが、後述する平準化によってそのバラツキが抑えられる。

この態様は、後述する実施形態における「第３の改良点：距離推定に適したデータ空間の分割」の記載に関連する。

[0020] さらに、各ハッシュテーブルは、各点を予め定められたｎ個のクラスタに

クラスタリングしてクラスタごとの代表点を算出し、それぞれのクラスタに属する各点からそのクラスタの代表点までの平均距離を表す分散が予め定められた閾値よりも小さくなるように各ビンの幅が決定されてもよい。このようにすれば、ビンの各点の分散が閾値以下になるように各ビンの幅を決定することにより、各ビンに登録される点の数を適切な範囲に平準化することができる。

この態様は、後述する実施形態における「第３の改良点：距離推定に適したデータ空間の分割」の記載に関連する。

[0021] さらに、前記探索範囲決定部は、各基底の方向におけるベクトルデータの分布から確率密度関数を求め、その確率密度関数を距離の重み付けに用いて前記推定値を決定してもよい。このようにすれば、確率密度関数を用いることによって複雑な分布に対しても適切な平準化が可能になる。

この態様は、後述する実施形態における「第４の改良点：確率密度関数に基づく距離推定」の記載に関連する。

また、前記探索範囲決定部は、クエリを中心として予め定められた探索半径Ｒの範囲内に代表点のある領域を前記探索領域とするものであってもよい。このようにすれば、探索半径Ｒを予め定めておくことによって、探索領域を決定することができる。

この態様は、後述する実施形態における「第５の改良点：クエリ周辺のデータ密度を考慮した探索半径の拡張」の記載に関連する。

[0022] あるいは、前記探索範囲決定部は、クエリを中心として探索半径Ｒの範囲内に代表点のある領域を前記探索領域とし、その探索領域に含まれる点の数が予め定められた数に達するまで探索半径Ｒを漸次大きくするものであってもよい。このようにすれば、クエリが入るバケットおよびその周辺のバケットに登録されたデータに多少があっても、探索領域に含まれる点の数、即ち最近傍点の候補数が予め定められ数になるように探索領域を決定するので、近似最近傍探索の精度を安定させることができる。また、探索領域内の各点との距離計算に要する処理時間を略一定にすることができる。

この態様は、後述する実施形態における「第５の改良点：クエリ周辺のデータ密度を考慮した探索半径の拡張」の記載に関連する。

[0023] さらに、前記データベース格納部は、主成分分析により基底が決定されたν次元空間に各点が射影されてなり、前記探索範囲決定部は、前記クエリと各代表点との距離の各基底方向における距離成分をそれぞれ算出して前記推定値とし、各距離成分の算出の過程において、各距離成分の和が定められた探索半径Ｒ内であることを制約条件とし、前記主成分分析において大きな固有値を有する基底の順に各領域の代表点が半径Ｒ内にあるかを判断する分枝限定法を適用して前記探索領域になり得ない領域を刈り込み、前記探索領域を決定してもよい。このようにすれば、分散の大きな基底方向から順に代表点が半径Ｒ以遠の領域を刈り込んで前記探索領域から除外するので、各領域が探索半径Ｒ内にあるかを総当たり的に判断する場合に比べると短時間で探索領域を決定することができる。

この態様は、後述する実施形態における「第１の改良点：ハッシュに基づく点対バケットの距離推定」の記載に関連する。

[0024] 前記データベース格納部は、各点にν個（νは２以上の整数）のハッシュ関数をそれぞれ適用してν個のインデックスを算出し、ν個の基底で構成され、各基底に沿って複数のビンに分割されかつ各インデックスによって何れかのビンが特定されるν次元空間に各点を射影することにより生成され、各ビンへの各点の登録がハッシュテーブルを用いて表されたものであってもよい。

この態様は、後述する実施形態における「第１の改良点：ハッシュに基づく点対バケットの距離推定」の記載に関連する。

[0025] 前記データベース格納部は、ν次元空間に各点が射影され、前記多次元ハッシュテーブルは、前記ν次元空間を張るν本の直交基底のうちＰ本の直交基底が張る部分空間をＭ組選択し、各部分空間にk-means法を用いてその部分空間が分割され、かつ、各領域内の分散が大きい部分空間ほど分割の数が大きく設定されてなり、前記検索範囲決定部は、各基底の方向におけるベクト

ルデータの分布から求めた確率密度関数に従って各領域に存在する点とクエリとの二乗距離の推定誤差が各基底方向においてそれぞれ最小化されるように前記推定値を決定してもよい。

この態様は、後述する実施形態における「第６の改良点」の記載に関連する。

この発明の好ましい態様は、ここで示した複数の態様のうち何れかを組み合わせたものも含む。

[0026] 《この発明を実行するハードウェアの構成例》

ここでは、近似最近傍探索の具体的な応用の一態様として画像認識装置について述べる。

図２０は、この発明の近似最近傍探索装置の応用例としての画像認識装置示すブロック図である。この発明に係る近似最近傍探索は、図２０に示す画像認識装置上でコンピュータが記憶装置等のハードウェア資源を用いつつ画像データを処理することによって実現される。画像認識装置のハードウェアは、例えば、ＣＰＵと、前記ＣＰＵが実行する処理手順を示すプログラムを格納したハードディスク装置などの記憶装置、前記ＣＰＵにワークエリアを提供するＲＡＭ、データを入出力する入出力回路などから構成される。より具体的には、上記構成を有するパーソナルコンピュータによって画像認識装置が実現されてもよい。あるいは、機器に組み込まれ、上記構成を有するマイクロコンピュータから構成されてもよい。

[0027] 図２０で、例えばデジタルカメラで撮影された画像のデータが、通信あるいは記憶媒体を介して画像認識装置に入力される。特徴点抽出部１１は、入力された画像のデータに含まれる対象物のパターンから公知の手法を用いて特徴ベクトルを抽出するブロックである。図２０の構成では、特徴ベクトルは画像の局所的な特徴を表し、１つの画像から複数箇所の局所特徴をそれぞれ表す複数の特徴ベクトルを抽出するものとしている。なお、別の態様として、一つの画像を一つの特徴ベクトルで表す手法もある。

探索範囲決定部１３は、各特徴ベクトルにハッシュ関数を適用して後述す

るハッシュテーブル１５ｈのインデックスを算出し、ハッシュテーブル１５ｈが有するビン（複数のハッシュ関数を用いる場合はバケット）を参照する処理を行う。

[0028] 画像データベース１５に画像のデータを登録するとき、前記ＣＰＵはその画像を識別する画像ＩＤを付して画像データベース１５に格納し、さらにその画像から抽出された特徴ベクトルを画像ＩＤと関連付けてハッシュテーブル１５ｈに登録する。詳細には、前記ハッシュ関数を適用して各特徴ベクトルを複数のビンの何れかに分類し、そのビンに登録しておく。つまり、各特徴ベクトルにハッシュ関数を適用してインデックスを算出し、各特徴ベクトルが前記インデックスに応じたビンまたはバケットに登録された画像データベース１５を作成する。登録時に用いるハッシュ関数は、探索範囲決定部１３がインデックスの算出に用いるハッシュ関数と同一である。

[0029] 以上のようにして画像データベース１５に画像が登録された後、クエリとして画像のデータが入力されると、画像認識装置は、画像データベース１５に格納された画像の中からクエリの画像に最も近い画像を探索し、認識結果として出力する。ここで、最も近い画像の探索は、特徴ベクトルどうしを比較し、クエリの各特徴ベクトルについて、最近傍のベクトルを探索することにより実現される。この最近傍ベクトルの探索に近似最近傍探索が適用される。

[0030] クエリとしての画像が与えられると、特徴点抽出部１１として前記ＣＰＵは、クエリから特徴ベクトルを抽出する。以下、抽出された特徴ベクトルをクエリベクトルと呼び、ハッシュテーブル１５ｈに登録されている特徴ベクトルを参照ベクトルと呼ぶ。

探索範囲決定部１３として前記ＣＰＵは各クエリベクトルに前述したハッシュ関数を適用してインデックスを得る。そして前記ＣＰＵは、得られたインデックスで特定されるハッシュテーブル１５ｈのビンを参照し、そのビンに登録されている参照ベクトルをクエリベクトルに対する最近傍の候補とする。このようにクエリベクトルにハッシュ関数を適用してビンを参照し、そ

のビンに登録された参照ベクトルを最近傍の候補とする処理は、近似最近傍探索の一段目の処理に該当する。最近傍候補、即ち距離計算の対象を絞り込む処理である。探索範囲決定部１３およびハッシュテーブル１５ｈは、近似最近傍探索の一段目の処理を具現化する構成といえる。ハッシュ関数は、参照先のビンが最近傍である確率が高い参照ベクトルを含む一方で、各ビンに登録される参照ベクトルの数が少なくなるようにバランスを考慮して、予め定められる。

[0031] 参照先のビンに唯一つの参照ベクトルが登録されている場合、前記ＣＰＵは、その参照ベクトルに対応付けられた画像ＩＤを認識結果の候補とする。一方、参照先のビンに複数の参照ベクトルが登録されている場合、最近傍点決定部１７として前記ＣＰＵは、それらの参照ベクトルのうちで最近傍ベクトルを決定する。詳細には、前記ＣＰＵは、クエリベクトルと参照先のビンに登録された参照ベクトルとの距離計算をそれぞれ行う。そして、クエリベクトルに最も距離の近い参照ベクトルを最近傍ベクトルとして決定する。その参照ベクトルに関連付けられた画像ＩＤを認識結果の候補とする。最近傍点決定部１７は、近似最近傍探索の二段目の処理を具現化する構成に該当する。

[0032] 前記ＣＰＵは、一つのクエリが与えられると、そのクエリから抽出された複数のクエリベクトルのそれぞれについて最近傍の参照ベクトルを求め、その参照ベクトルに関連付けられた画像ＩＤを認識結果の候補とする。

投票部１９として前記ＣＰＵは、各クエリベクトルについて認識結果の候補とされた画像ＩＤの投票を行う。投票の際に各画像ＩＤ（画像１から画像ｎ）の得票数を記憶する投票テーブル２１が設けられている。このように投票による多数決処理を経て認識結果を得る利点は、いくつかのクエリベクトルが誤った画像ＩＤに対応付けられても、最終的に正しい認識結果が得られる可能性が高いことである。画像の撮影、特徴ベクトルの抽出の各過程は幾何学的歪み、解像度変換、明暗の変化等に伴う誤差要因を含んでいる。また、近似最近傍探索それ自体が処理時間とのトレードオフとして探索誤りを許

容するために誤差要因を含む。よって、全てのクエリベクトルが正しい画像ＩＤに対応付けられるとは限らない。高精度の画像認識を実現するうえでこのような多数決処理は有効である。

画像選択部２３として前記ＣＰＵは、投票テーブル２１を参照して最大得票数を得た画像ＩＤに係る画像を最終的な認識結果とする。

[0033] 以上が画像認識装置の構成である。このうち、クエリベクトルにハッシュ関数を適用する探索範囲決定部１３、特徴ベクトルを体系化して格納するハッシュテーブル１５ｈ、同一ビンに登録された参照ベクトルのうちで最近傍ベクトルを決定する最近傍点決定部１７は、近似最近傍探索に係る構成であり、最近傍探索装置を構成する要素といえる。なお、図２０の画像認識装置は認識結果として画像を出力するものであるため、例えば最近傍点決定部１７は最近傍ベクトルに関連付けられた画像ＩＤを出力する。このうち、近似最近傍探索装置としての構成部分は、入力された特徴ベクトルに対してハッシュテーブルに登録された特徴ベクトルの中から最近傍の特徴ベクトルを決定して出力するものである。特徴ベクトルに関連付けて画像ＩＤを格納する構成部分および最近傍ベクトルに関連付けられた画像ＩＤを出力する構成部分は含まない。なお、図２０の画像認識装置は最近傍探索の応用例であって、最近傍探索装置が扱うデータは特徴ベクトルに限定されるものではない。

[0034] 《従来の代表的な近似最近傍探索手法》

次に、従来の代表的な近似最近傍探索手法について述べる。これにより、後述するこの発明の実施形態がより理解しやすくなるであろう。

〈１．ＡＮＮ〉

木構造を用いる手法の中で最も代表的なものの一つがApproximate Nearest Neighbor（ＡＮＮ）である。ＡＮＮは２分木をベースとしている。木の構築ではデータ空間を階層的に２等分していき、葉に入る点が１つになるまで分割を繰り返す。クエリが与えられると、木を辿り、到達した葉に登録されているデータと距離計算を行う。その距離がｒであるとすると、分割された各領域の最も近いところがクエリから半径ｒ／（１＋ε）に入る領域を探索領

域とする。εは近似度パラメータであり、ε＝０であれば、ｒより近い点が存在する可能性のある領域を全て探索するので、必ず真の最近傍点を得ることができる。

[0035] 〈２．ＦＬＡＮＮ〉

Fast Library for Approximate Nearest Neighbors（ＦＬＡＮＮ）は与えたデータベースに合った近似最近傍探索手法とそのパラメータチューニングを提供するライブラリである。このライブラリには、提案されている手法の中で高い性能を持つrandamized kd-tree（例えば、非特許文献１２参照）と階層的k-means（例えば、非特許文献１３参照）に加えて全数探索が採用されている。

[0036] randamized kd-treeは複数の木から再近傍候補を選択する手法である。ＡＮＮのような通常のkd-treeを用いた方法では、着目するデータの要素を順に変えながら空間を２分割していくことで木構造を構成する。このとき、高次元データに対しては高い精度を得ようとすると構造的に最近傍点が含まれる確率の低い葉にまで探索範囲広げる必要がある。しかし、これでは木の巡回に要する処理時間や多くの無駄な距離計算をすることになる。そこで、randamized kd-treeでは主成分分析を行い、距離計算への寄与度が高い上位Ｄ次元の基底にのみ着目してkd-treeを構成する。このとき、各階層での基底選択をランダムに行い複数の木を構成する。従って、探索時には一つ一つの木の探索範囲が小さくても複数の木を辿ることで高い精度を確保することが可能になり、通常のkd-treeよりも高い性能を得ることができる。

階層的k-meansはその名の通り、各ノードで自身に所属する点をk-meansによってクラスタリングし、各階層で空間をクラスタごとに分割する。

[0037] 〈３．ＬＳＨ〉

Locality Sensitive Hashing（ＬＳＨ）はハッシュを利用した近似最近傍探索手法の中で最も代表的な手法の一つである。ここではＬＳＨの中でもこの発明に関連する、ベクトル空間で用いることができるＬＳＨ（例えば、非特許文献８参照）について述べる。ＬＳＨは、複数のハッシュ関数を利用し

て、クエリの近傍にあると考えられる点を選出し、それらの点に対してのみ距離計算を行う。即ち、データ空間をランダムに生成された複数の基底の方向に等間隔に分割することで、空間をバケットと呼ばれる領域に分割してインデクシングを施す。

[0038] 図２１は、従来の近似最近傍探索の一手法であるＬＳＨの説明図である。図２１（ａ）は、データ空間がランダムに生成された２つの基底a_1およびa_2の方向に沿ってそれぞれハッシュテーブルのビンで等分割された様子を示している。軸a_1に沿って分割された各領域は基底a_1に係るハッシュ関数h_{j1}によってインデクシングされたビンであり、軸a_2に沿って分割された各領域は基底a_2に係るハッシュ関数h_{j2}によってインデクシングされたビンである。各軸にインデックスの値を示している。そして、それら２種類のビンが交差するセル状の１つ１つの領域、即ち２次元ハッシュテーブルの各次元のビンが交差する積領域がバケットである。各バケットの数値は、ハッシュ関数h_{j1}およびh_{j2}のインデックスの値を示している。

[0039] 探索時にはクエリと同じバケットに属する点を最近傍候補とする。しかし、これだけでは真の最近傍点を候補から漏らす可能性が高いため、この処理を数回繰り返すことで候補を増やして精度を上げる工夫をしている。図２１（ｂ）は３回の射影によって得られた探索領域の様子を表す。

[0040] 非特許文献８のＬＳＨでは次式のようなハッシュ関数群を用いる。

[数1]

$$g(\boldsymbol{x}) = \{H_1(\boldsymbol{x}), H_2(\boldsymbol{x}), \ldots H_L(\boldsymbol{x})\} \quad (1)$$

$$H_j(\boldsymbol{x}) = \{h_{j1}(\boldsymbol{x}), h_{j2}(\boldsymbol{x}), \ldots h_{jk}(\boldsymbol{x})\} \quad (2)$$

$$h_{ji}(\boldsymbol{x}) = \left\lfloor \frac{\boldsymbol{a}_{ji} \cdot \boldsymbol{x} + b_{ji}}{W} \right\rfloor \quad (3)$$

ただし、ｘは任意の点、a_iは各次元の要素の値がガウス分布から独立に選ばれたベクトル、Ｗはハッシュ幅であり、b_iは区間［０，Ｗ］から一様に選ばれた実数である。そして、最近傍候補はクエリｑに対して∃（g_j（ｑ）＝g_j（ｐ））ｊ＝１，…，Ｌとなる点ｐである。ＬＳＨが近似最近傍点を求める

ことができるのは局所性に鋭敏な（Locality Sensitive）ハッシュ関数を用いているためである。局所性に鋭敏なハッシュ関数とは、距離が近い点同士は同じハッシュ値（インデックス）を取る確率が高く、距離が遠い点同士は同じハッシュ値を取る確率が低いハッシュ関数である。

[0041] 式（２）のごとく、ＬＳＨではハッシュ関数h_{ji}をｋ個組み合わせてハッシュ関数群H_jを作る。これは、ｋ次元のハッシュテーブルに対応する。図２１（ａ）の灰色の領域は、ｋ＝２（射影空間が２次元）のときクエリにh_{j1}とh_{j2}の２つのハッシュ関数を適用して求めたビンの積領域（共通の領域）、言い換えるとクエリにハッシュ関数群H_jを適用して求めたバケットである。このバケットが一つの関数群H_jに係る距離計算の対象領域である。このようなハッシュ関数群H_jをＬ個（Ｌ組）作り、最終的にＬ個の領域を組み合わせた領域をクエリとの距離計算の対象領域とする。図２１（ｂ）はＬ＝３の場合を示している。ＬＳＨは、上記の手順で距離計算の対象を削減して処理を高速化する。

ＬＳＨはデータの分布に依らないランダムな斜交基底に射影を行うため、データ空間における距離を射影空間が保持するという観点から効率的でない。

[0042] 〈４．ＳＨ〉

バイナリ符号のハッシュ関数を用いた手法の中でも代表的なSpectral Hashing（ＳＨ）の概要について述べる。ＳＨはハッシュを用いたもの中で良い性能が得られるといわれている手法である。ＳＨはデータ空間の主成分を上からいくつか選択し、ハミング空間への射影を行う。そして、射影されたハミング空間における距離（ハミング距離）が閾値以下のものを最近傍候補とする。即ち、上位の主成分基底にのみ着目して各サンプルをバイナリ符号に変換し、クエリとのハミング距離によって最近傍候補を選択する。ＳＨの符号化はデータ空間の一様分布を仮定し、分割された領域がなるべく直方体に近い形で分割されるように空間を分割し、各バケットにバイナリ符号を与える。

[0043] 図２２は、従来の近似最近傍探索の一手法であるＳＨの説明図であり、データ空間が２つの主成分基底ｐｖ$_1$およびｐｖ$_2$からなる２次元のハミング空間に射影された様子を示したものである。各軸にインデックスをバイナリ符号で示している。各バケットの符号は、軸ｐｖ$_1$および軸ｐｖ$_2$のインデックスを組み合わせたものである。クエリは符号111のバケットに属する。灰色の領域はハミング距離の上限を１とした場合のクエリに対する探索領域を表したものである。即ち、符号111のバケットに加えて、符号111と１つの符号のみが異なる符号110、101、011のバケットが探索領域となる。

ＳＨはデータ空間を主成分基底に射影するため、射影後も元の距離が保持されやすいといえるが、射影した空間での距離がハミング距離で表されるため、ユークリッド距離との誤差が生じる。例えば、図２２でクエリが射影されるバケット111から遠い011の領域が最近傍候補となるといった問題がある。

[0044] 〈５．佐藤らの手法〉

佐藤らの手法（例えば、非特許文献１１参照）は、クエリが入ったバケットの重心から各バケットの重心までの距離（バケット距離）の概算値を求めてクエリから各点の距離の推定値とし、クエリからの距離が小さい点を探し出そうという手法である。

佐藤らの手法では、データ空間を任意の正規直交基底に対して共通の分割幅で等分した空間に射影し、これを多次元ハッシュによって表現する。この処理は、データをスカラー量子化することに等しく、射影空間上の距離は、元のデータ空間の距離（真の距離）をよく反映したデータ構造となっている。探索時には、クエリが属するバケットと各点が属するバケットとの距離を求めることで、クエリを中心とする近似的な超球領域から最近傍候補を抽出する。

[0045] 図２３は、従来の近似最近傍探索の一手法である佐藤らの手法において各バケットの距離を定める手順を示す説明図である。図２３で、射影空間は縦軸と横軸の２つの正規直交基底で分割されている。即ち、ハッシュの次元数

ν＝２である。縦横の各基底に対して共通の分割幅でビンに分割されている。縦軸と横軸に付された数値はインデックスである。セル状になっている一つ一つの区画がビンの積領域としてのバケットである。各バケットに振られている数字の並び１１～３３は、バケットを作るビンのインデックスを組み合わせたものである。この数字の並びはデータ空間内のバケットの位置を示す位置ベクトルとして考えることができる。このインデックスの並びつまりバケットの位置ベクトルから、バケット間の距離Ｄが定義される。

[0046] クエリが入ったバケットが分かれば、そのバケットから他の何れかのバケットまでの距離はバケットのインデックスを用いて知ることができる。よって、最近傍点の探索は、バケット距離の小さいバケットから順にバケットに含まれるデータを参照していけばよい。このようにすれば、クエリを中心とした超球領域の探索が実現できる。そして、クエリからの距離の概算値が小さい点のみを距離計算対象とすることができる。

[0047] いま、ｘを任意の点、Ψ_iを正規直交基底、Ｗを分割幅（ビンの幅）とすると、ν次元ハッシュ関数Ｈは次のようになる。

[0048] [数2]

$$H(\boldsymbol{x}) = \{h_1(\boldsymbol{x}), h_2(\boldsymbol{x}), \ldots, h_v(\boldsymbol{x})\} \tag{4}$$

$$h_i(\boldsymbol{x}) = \left\lfloor \frac{\Psi_i \cdot \boldsymbol{x}}{W} \right\rfloor \tag{5}$$

２点間の概算距離には射影空間上で任意の２点p_1，p_2がそれぞれ属するバケットのハッシュ値から求めたユークリッド２乗距離の期待値を用いる。データ空間の一様分布と各基底の独立を仮定すると、距離の期待値は次のようになる。

[0049] [数3]

$$E\left[\sum_{i=1}^{v}(p_1 \cdot \Psi_i - p_2 \cdot \Psi_i)^2\right]$$

$$= \left[\sum_{i=1}^{v}\{h_i(p_1) - h_i(p_2)\}^2 + \frac{v}{6}\right] W^2$$

上式から、距離の比較には射影空間における距離（バケット距離）を用いれば十分であり、これを概算距離として探索を行う。なお、２乗距離は一例である。

[0050] ここで、Ｂ（ｐ）を任意の点ｐが属するバケットの重心であるとすると、２点p_1，p_2がそれぞれ属するバケットのバケット距離は次のように表される。

[数4]

$$D(B(\boldsymbol{p}_1), B(\boldsymbol{p}_2)) = \sum_{i=1}^{v} (h_i(p_1) - h_i(p_2))^2 \qquad (6)$$

[0051] 図２３を例に探索領域の選択について説明する。図２３は、クエリにハッシュ関数を適用したときのビンのインデックスが〈２，２〉のときの例である。まず、クエリと同じインデックスのバケットに登録されている点と距離計算を行う。次に、バケット距離１のバケットを探索する。図２３ではインデックス〈１，２〉、〈２，１〉、〈２，３〉、〈３，２〉のバケットである。これらのバケットを順に探索していく。十分な数のバケットを探索したと判断すれば、探索を終了する。一方、まだバケットの数が不十分であると判断すればさらに遠いバケットに探索範囲を広げていく。図２３では、インデックス〈１，１〉、〈３，１〉、〈３，３〉、〈１，３〉のバケットである。

[0052] 図１は、従来の佐藤らの手法における距離推定の一例を示す説明図である。図１で、縦軸と横軸は、図２３と同様に二次元の正規直交基底である。星印はクエリを表す。図２３と異なり、各軸の数字はクエリを基準（原点）としたときの左右方向および上下方向におけるバケット距離の重みを表している。図１では、重みはクエリの入ったビンからのユークリッド２乗距離としている。また、バケット内の数字は推定されたバケット間の距離を表す。バケット間の距離は、左右方向および上下方向の重みの和である。探索したバケットの数が十分か否かの判定として、探索時には探索半径Ｒを与え、Ｄ（Ｂ（ｑ），Ｂ（ｐ））≦Ｒを満たす点ｐを最近傍候補とする。仮に探索半径

をＲ＝２とすれば、推定距離が２以下の９つのバケットが最近傍候補として選択される。Ｒはクエリに応じて決めてもよい。

[0053] この手法では、データ空間の次元数に対して射影空間の次元数が小さいと概算距離の精度が低下する。しかし、ハッシュサイズはハッシュの次元数νに対して指数関数的に大きくなるため、高次元データに対して精度を保持するためには膨大なハッシュサイズが必要となる。高次元データに対してハッシュの次元数νを大きくすることは難しく、結果的に高次元データに対して十分な概算距離の精度を得ることができない。また、ハッシュサイズが大きくなりすぎると、最近傍探索の精度を維持するために多くのバケットを参照する必要があり、最近傍候補の抽出処理に多くの時間がかかる。

〈６．IVFADCとＩＭＩ〉

Inverted File with Asymmetric Distance Calculation(IVFADC)やその改良手法であるInveted Multi-Index(ＩＭＩ)は、k-means法によってデータ空間を粗く量子化（粗量子化）することで、空間を分割（インデクシング）する。このとき粗量子化により得られる各クラスター（分割された領域）の代表値（セントロイド）の集合をＣとし、クラスターの総数、即ちセントロイドの総数を|Ｃ|＝Ｇとする。このとき、クエリから各領域に属する点までの距離の期待値（推定距離）はその領域のセントロイドまでの距離である。従って、クエリに近いセントロイドを求め、その領域に属する点を最近傍候補とすれば効率がよい。

１．IVFADCの最近傍候補選択

IVFADCは粗量子化に単純なベクトル量子化を用いる（例えば、H. Jegou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," IEEE Trans. TPAMI, vol.33, no.1, pp.117-128, 2011参照）。最近傍候補の選択精度を向上させるには空間を細かく分割する（Ｇを大きくする）必要がある。しかし、最近傍候補選択の計算コストはＯ（Ｇ）となり、大きな値をとることができない。従って、最近傍候補選択の精度が十分に得られないという問題がある。

２．ＩＭＩの最近傍候補選択

そこでIVFADVの改良手法として提案されたのがＩＭＩである（例えば、A. Babenko and V. Lempitsky, "The inverted multi-index," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.3069-3076, 2012参照）。粗量子化ではベクトルｘを２つの部分ベクトルU^1（ｘ），U^2（ｘ）に分割するプロダクト量子化を行う。得られた部分セントロイドの集合をC^1，C^2、その要素数$|C^1|=|C^2|=$をｇとする。セントロイドの集合は$C=C^1\times C^2$で得られ、その数は$G=g^2$となる。C^1のｉ番目とC^2のｊ番目の要素を並べてできるセントロイドを$c_{ij}=\{c_i^1, c_j^2\}$と定義すると、クエリｑとc_{ij}に属する点までの推定二乗距離F_{ij}（ｑ）は、クエリから第ｍ部分セントロイドのｉ番目の要素までの距離を

[数5]

$$f_i^m(\boldsymbol{q}) = D^2(U^m(\boldsymbol{q}), \boldsymbol{c}_i^m)$$

として、次式で表される。

[数6]

$$\begin{aligned} F_{ij}(\boldsymbol{q}) \equiv & D^2(\boldsymbol{q}, \boldsymbol{c}_{ij}) \\ = & D^2\left(U^1(\boldsymbol{q}), \boldsymbol{c}_i^1\right) + D^2\left(U^2(\boldsymbol{q}), \boldsymbol{c}_j^2\right) \\ = & f_i^1(\boldsymbol{q}) + f_j^2(\boldsymbol{q}) \end{aligned}$$

従って、セントロイドをクエリに近い順に選択する問題は、２つの部分距離リスト

[数7]

$$\{f_i^1\}, \{f_j^2\}$$

の中から１つずつ選択して、その和F_{ij}が小さくなるｉとｊの組み合わせを探索する問題に帰着する。ＩＭＩではこの問題を Multi-Sequenceアルゴリズムと呼ばれる組み合わせ探索法を用いて解く。このアルゴリズムは、ある時点で最も距離が小さい添え字の組み合わせを生成する毎に、次に小さい距離の組み合わせを生成する可能性がある添え字の組としての添え字候補を追加し

、その中から次の組み合わせを選択する処理を繰り返す。そして、得られた最近傍候補数がＬ点に達した時点で探索を終了する。

空間の分割数Ｇが等しいとき、プロダクト量子化はベクトル量子化に比べて精度は悪くなる。しかし、プロダクト量子化を用いることでMulti-Sequence アルゴリズムが適用でき、計算コストが

[数8]

$$O(g \log g) = O(G^{1/2} \log G)$$

で最近傍候補を得られるため、探索の高速化が実現できる。上述の文献"The inverted multi-index,"では、IVFADCに比べて高速に解を得られることが報告されている。

以上、従来の代表的な近似最近傍探索手法について述べた。

[0054] 《この発明に係る距離推定の手法》

以下、図面を用いてこの発明をさらに詳述する。なお、以下の説明は、すべての点で例示であって、この発明を限定するものと解されるべきではない。

近似最近傍探索が高精度かつ高速であるためには、最近傍候補の抽出において真の最近傍点を漏らさずに候補を減らし、またこの処理自体が高速であることが重要となる。このとき、クエリを中心とする超球領域を探索して最近傍候補とすることができれば真の最近傍点を漏らすことがなく理想である。しかし、データ空間が高次元である場合、距離計算を行わずにこれを実現するのは容易ではない。ハッシュのインデックスに基づく概算距離が元の空間での距離をうまく保持していれば、インデックスに基づいてクエリからの距離を推定できる可能性がある。しかし、従来手法の多くは射影後の空間における距離が元の空間における距離を十分に保持することができず、推定距離が真の距離の大小関係を十分に保持することができない。その場合、探索の精度を上げるためには多くの最近傍候補を確保する必要があり、高速に解を得ることができない。

[0055] そこで、発明者らは、概算距離が真の距離の大小関係をよく反映している

佐藤らの手法に着目し、さらに改良された近似最近傍探索手法を提案する。これによって、従来の手法と比べて高精度かつ高速な距離推定が可能になり、ひいては近似最近傍探索の処理全体の高速化が実現される。

この発明は、近似最近傍探索の一段目の処理における最近傍候補の選択精度を向上させ、最近傍候補の選択を高速に実行するために、距離推定の手法および適応的な探索範囲の決定手法に特に着目している。

[0056] 佐藤らの手法における距離推定に対する第１の改良点は、ハッシュに基づき点対バケットで距離を推定する手法である。第２の改良点は、分割されたハッシュテーブルを用いる手法である。そして、第３の改良点は、距離推定に適した空間分割の手法である。第４の改良点は、確率密度関数に基づく距離推定の手法である。さらに、第５の改良点は適応的探索範囲に係るものであり、クエリ周辺のデータ密度を考慮した探索領域の拡張である。各手法は、単独で適用することもできるが、全部または一部を組み合わせて適用することができる。以下、各手法の詳細について述べる。併せて、各手法の有効性を示す実験例について述べる。

[0057] 以下の説明では、前述した改良点のうち第１の点対バケットの距離推定および第２のハッシュテーブルの分割を適用した近似最近傍探索手法をBucket Distance HashingあるいはＢＤＨと呼ぶ。さらに、ＢＤＨに第３の改良点である距離推定に適した空間分割を組み合わせた手法をk-means ＢＤＨと呼ぶ。そして、さらにk-means ＢＤＨに第４の改良点である確率密度関数に基づく距離推定を組み合わせた手法をk-means ＢＤＨ Ｐと呼ぶ。また、k-means ＢＤＨ Ｐに第５の改良点である適応的探索範囲を組み合わせた手法をk-means ＢＤＨ ＰＣと呼ぶ。

[0058] 〈第１の改良点：ハッシュに基づく点対バケットの距離推定〉

１つ目の改良された手法は、主として距離推定の精度向上に係るものである。佐藤らの手法ではクエリとデータのそれぞれが属するバケット間の距離、すなわちバケット対バケットの距離を推定した。ここでは厳密なクエリの位置から各データが属するバケットへの距離、すなわち点対バケットの距離

を推定することにより、同じデータ構造のまま距離推定の精度を向上させる手法を提案する。

[0059] この手法は、ハッシュ関数として佐藤らの手法と同じ式（５）を用いる。つまり、２点ｐ$_1$，ｐ$_2$についてｐ$_1$の座標とｐ$_2$と同じバケットに属する点の座標の期待値を計算して代表点を定める。具体的には、代表点の座標がバケットに属する点の座標の期待値になるように代表点を定める。すなわち、代表点はバケットの重心になる。そして、ｐ$_1$の座標とｐ$_2$が属するバケットの代表点の座標の２点間のユークリッド距離を計算し、これを概算距離とする。まず、基底ｉ方向のユークリッド２乗距離の期待値を考えると次のようになる。

[数9]

$$
\begin{aligned}
&E[(\boldsymbol{p}_1 \cdot \Psi_i - \boldsymbol{p}_2 \cdot \Psi_i)^2] \\
=&\frac{1}{W}\int_{h_i(p_2)W}^{(h_i(p_2)+1)W} (\boldsymbol{p}_1 \cdot \Psi_i - x)^2 dx \\
=&\left[\left\{\frac{\boldsymbol{p}_1 \cdot \Psi_i}{W} - \left(h_i(\boldsymbol{p}_2) + \frac{1}{2}\right)\right\}^2 + \frac{1}{12}\right] W^2
\end{aligned}
$$

また，各基底が独立であるという仮定からv 次元空間におけるユークリッド２乗距離の期待値は次のようになる。

[0060] [数10]

$$
\begin{aligned}
&E\left[\sum_{i=1}^{v}(\boldsymbol{p}_1 \cdot \Psi_i - \boldsymbol{p}_2 \cdot \Psi_i)^2\right] \\
=&\left[\sum_{i=1}^{v}\left\{\frac{\boldsymbol{p}_1 \cdot \Psi_i}{W} - \left(h_i(p_2) + \frac{1}{2}\right)\right\}^2 + \frac{v}{12}\right] W^2
\end{aligned}
$$

上式から、距離の比較には次式を用いれば十分であることが分かり、これはｐ$_1$とｐ$_2$が属するバケットの代表点としての重心との距離計算をしていることに等しく、これを概算距離として探索を行う。なお、ユークリッド２乗距離は一例であって、この発明の本質はこれに限定されるものではない。

[0061] 点p_1とp_2が属するバケットB（p_2）の間の距離、つまり点対バケットの距離は次のように表される。

[数11]

$$D(\boldsymbol{p}_1, B(\boldsymbol{p}_2)) = \sum_{i=1}^{v} \left\{ \frac{\boldsymbol{p}_1 \cdot \Psi_i}{W} - \left(h_i(\boldsymbol{p}_2) + \frac{1}{2} \right) \right\}^2 \quad (7)$$

式（7）で、（h_i（p_2）＋1／2）はバケットB（p_2）の重心のΨ_i方向の座標を表しており、式（7）の距離はクエリとバケット重心の距離に等しい。クエリの位置が特定されている分、式（6）よりも精度の高い距離推定を実現している。推定距離の誤差分散は、バケット対バケットの距離を推定する佐藤らの手法に比べて1／W^2となる。

[0062] ここで後の説明のために式（7）の距離をW倍して次式のように置き直す。

[数12]

$$D(\boldsymbol{p}_1, B(\boldsymbol{p}_2)) = \sum_{i=1}^{v} \mathrm{BD}_i(\boldsymbol{p}_1, B(\boldsymbol{p}_2)) \quad (8)$$

$$\mathrm{BD}_i(\boldsymbol{p}_1, B(\boldsymbol{p}_2)) = \left\{ \boldsymbol{p}_1 \cdot \Psi_i - \left(h_i(\boldsymbol{p}_2) + \frac{1}{2} \right) W \right\}^2$$

ここで、BD_i（p_1，B（p_2））は、点p_1とバケットB（p_2）の重心との第i次元の基底方向における距離を表す。

この置き直しによって推定される距離がW倍されることになるが、最近傍候補の選択は相対的な距離に基づいて行われるため、結果は変わらない。

[0063] 図2は、この実施形態に係る距離推定の一例を示す説明図である。従来の佐藤らの手法における距離推定を示す図1に対応する図である。図1と同様に、図2中の星印はクエリを表す。バケットの中心間の距離を1とすれば、左右方向については、クエリは左のバケットの中心から0.6、右のバケットの中心から0.4の位置にあり、上下方向については上のバケットの中心から0.7、下のバケットの中心から0.3の位置にある。軸の数字（2.56，0.36など）は

左右方向および上下方向のバケット距離の重みを表し、バケット内の数字（3.05, 0.85など）は推定されたクエリと各バケットとの距離の重みを表す。重みはユークリッド２乗距離としている。

[0064] 図３は、この実施形態に係る距離推定の異なる一例を示す説明図である。図３でハッシュ関数の次元数ν＝２である。図２とは別の探索空間とクエリを表した例である。図３のクエリを中心とした円が探索半径の大きさを表しており、バケット中心が円内にあるバケットを探索領域とし、その探索領域を灰色で示している。探索時にはパラメータとして与えられた探索半径Ｒ以内のバケットを参照する。クエリの入ったバケットの中心ではなくクエリを中心とした円内（ハッシュの次元数νを任意の自然数に拡張した場合は超球）を探索範囲とするので、バケット対バケットの距離推定を行う佐藤らの方法に比べて高い精度で距離推定を行うことができる。

[0065] このように探索半径Ｒ以内のバケットを選択する処理は佐藤らの手法に比べて複雑になる。その理由は、佐藤らの手法では推定距離が整数に限定されていたのに対して、ＢＤＨでは実数であるため、推定距離を厳密に扱おうとすれば計算時間がかかる。この問題を避けるために、この発明では探索半径Ｒ以内のバケットを高速に特定するアルゴリズムを提案する。図１８は、この実施形態において探索半径Ｒ以内のバケットを高速に特定する分枝限定法に基づく２つのアルゴリズムを示している。「アルゴリズム１」および「アルゴリズム２」である。ただし、これらのアルゴリズム１、２は、いずれも特徴ベクトルの次元数をdとし、クエリを$q=\{q_1, q_2, \cdots, q_d\}$として、$BD_{ij}=BD_i(q, j)$とする。$BD_{ij}(p_1, B(p_2))$は、クエリとｊ番目のバケットの重心との第ｉ次元の基底方向における距離を表す。

[0066] ハッシュ関数の次元数がνの場合、クエリからあるバケットの代表点までの距離を求めるにはそれぞれν個の座標値が必要である。バケットの代表点の座標値はν個のハッシュ値、$H(x)=\{h_1(x), h_2(x) \cdots h\nu(x)\}$を使って求められる。これらが決まると，ν回の足し算によってバケットまでの距離がわかる。

ここでν個の基底は主成分分析によって選ばれ、固有値の降順に並んでいるため、データは（添字の番号が若い）上位の基底方向に大きな分散を持っている。従って、最も効率的な半径Ｒ以内のバケットの探索方法は以下で述べるように添字の番号順に評価することである。

[0067] ν個のハッシュ値のうち上位ｉ個のハッシュ値h_1（ｘ），h_2（ｘ）…h_i（ｘ）が既に決定したとすれば、残りν－ｉ個のハッシュ値を選択することによって半径Ｒ以内のバケットを探索することになる。ここで上位ｉ個のハッシュ値で計算される距離をD_iとおくと、ν－ｉ個の各ハッシュ関数において取りうる最小値を足して得られる距離mD_iとD_iの和が探索半径Ｒを超えれば、上位ｉ個のハッシュ値を使用する限りクエリから半径Ｒ以内という条件を満たす事はできない。このようにしてクエリから半径Ｒ以内に存在するバケットを順次探索する。

[0068] アルゴリズム１では１～３行目で、上位ｉ次元分のハッシュ値が決定したとき，残りのν－ｉ個のハッシュ値を自由に選べるとして，ν－ｉ次元分の距離の最小値mD_iを算出する。４～６行目ではクエリから最も近いバケットまでの距離$\Sigma_{b=1}^{\nu} \min_j BD_{bj}$が探索半径Ｒより小さければアルゴリズム２の関数を呼び出す。

[0069] アルゴリズム２は２つの引数を取る。何次元目のハッシュ値を決定するかを表すｉと既に確定したｉ－１次元分の距離Ｄである。１～７行目では、最後の基底(ν番目のハッシュ値を決める段階)に到達していなければ（１行目）、k_i個(第i基底の分割数、即ち第ｉ基底方向のバケットを構成するビンの数)のハッシュ値（インデックス）を試す（２行目）。BD_{ij}はハッシュ値ｊ（第ｊ番目のビン）を選択したときの第i基底における１次元分の距離（第ｉ基底方向の距離成分）であるので、Ｄ＋BD_{ij}＋mD_iはi個のハッシュ値が決まった段階で最も近いバケットまでの距離であり、これが探索半径Ｒよりも小さい場合のみ自分自身を再帰呼び出しして次の基底に進む（３～５行目）。９～１３行目では、最後であるν番目の基底に到達したときのみ実行され、探索半径以内のバケットがあればそのハッシュを引く。

[0070] 〈第２の改良点： ハッシュテーブルの分割〉

佐藤らの手法には高次元データに対して、高速な探索を行うことができないという問題がある。これはハッシュの次元数とハッシュテーブルに含まれるバケット数（以後ハッシュサイズ）の関係で生じる問題である。２点間の距離を低次元の部分空間に射影して計算する場合、部分空間の次元数が低い程推定された距離の精度は低くなる。従って、一定の推定精度を維持するためにはそれに応じた次元数の部分空間が必要になる。ハッシュを用いて距離推定する場合も同じで、推定距離の精度を維持するためにはデータの次元数に合わせてハッシュの次元数νを大きくする必要がある。しかし、ハッシュの次元数νを大きくするとハッシュサイズが膨大になり、メモリに収まりきらないといった事態に陥る。

[0071] 例えば、１つの基底の分割数、即ちビンの数をｓとすると、ハッシュテーブルとしてはｓ個のビンに対応するデータ格納領域をν組用意する必要がある。よって、ハッシュテーブルのサイズ（ハッシュサイズ）のオーダーはＯ（ｓν）となる。仮にビンの数ｓを最小の２に抑えたとしてもν＝30次元のハッシュを構成するには約10億のデータ格納領域が必要となる。

[0072] そこでデータの次元数の増大に対するハッシュサイズの増加を抑制するために、高次元のハッシュテーブルを分割し、低次元のハッシュテーブルから得られる推定距離を統合することによって高次元ハッシュの推定距離を求める手法を提案する。ν次元ハッシュテーブルをＭ個に分割する場合、次のようにν個のハッシュ関数をＭ個の組に分ける。

[数13]

$$H_j(\boldsymbol{x}) = \{h_{j1}(\boldsymbol{x}), h_{j2}(\boldsymbol{x}), \ldots, h_{jt_j}(\boldsymbol{x})\} \tag{9}$$

$$h_{ji}(\boldsymbol{x}) = \left\lfloor \frac{\Psi_{ji} \cdot \boldsymbol{x}}{w} \right\rfloor \tag{10}$$

ただし、ｊ＝１，…，Ｍ、かつ$\Sigma_j t_j = \nu$ある。即ち、H_j（ｘ）はＭ組に分割されたハッシュ関数の各組のハッシュ関数群であり、分割されたハッシュ関数H_j（ｘ）の次元数の総和はνである。

[0073] そして、クエリｑから任意の点ｐへの推定距離を

[数14]

$$D(q,B(p))=\sum_{j=1}^{M}D_j(q,B_j(p)) \tag{11}$$

で表すことができる。これはν次元ハッシュによって求められる推定距離に等しい。ここで距離計算の対象となるのはいずれかのハッシュテーブルでクエリから探索半径Ｒ以内に存在した点である。距離計算対象の点がｊ番目のハッシュテーブルでクエリから探索半径Ｒ以内に存在した場合はその距離をD_j（ｑ，B_j（ｐ））とし、存在しなかった場合はＤｊ（ｑ，B_j（ｐ））＝Ｒとする。ハッシュテーブルを分割する利点は、同じ次元数のハッシュを表現する場合でも１つのハッシュテーブルを用いる場合に比べて飛躍的にハッシュサイズが小さくなることにある。一つの基底方向にそれぞれｓ分割されている場合を考えると、１つのハッシュテーブルによってν次元ハッシュを表現する場合、ハッシュサイズはＯ（ｓ$^{\nu}$）あるのに対し、Ｍ個のハッシュテーブルに分割してν次元ハッシュを表現する場合、分割された１つのハッシュテーブルのサイズはＯ（ｓ$^{\nu/M}$）となり、Ｍに対して指数関数的に減少することが分かる。ハッシュ全体としてもその高々Ｍ倍に留まる。従って、ハッシュテーブルを分割して多次元ハッシュを表現することにより、高次元データに対しても最近傍候補の抽出速度を落とすことなく、距離推定の精度を向上させることができる。

[0074] 〈第１、第２の改良点に係る実験例〉

前述の第１および第２の改良点について、その有効性を確認する実験を行った。以下に実験の内容とその結果を記す。

〈１．予備実験〉

図４は、従来の佐藤らの手法の多次元ハッシュの次元数νを変化させたときの、精度と処理時間の関係を示すグラフである。ここで用いたデータは64次元または128次元であり、図４（ａ）は64次元、図４（ｂ）は128次元の場合を示す。1000万点の正規分布に基づく人工データおよびクエリは、どちら

も同じ条件で生成した2000点である。用いた計算機はCPUがOpteron（tm）6174（2.2GHz）、メモリは256[GB]であり、実験はシングル・コアで行った。

人工データにおいては佐藤らの手法と図示していないＢＤＨ共にν＝24が最も精度と処理時間の関係が良かったので、以降、本節での人工データを用いた実験においてはν＝24とする。

[0075] 〈２．実験〉

本節では提案したＢＤＨの性能を評価するため、前節で紹介した従来手法とこの発明の比較実験を行う。計算機は予備実験と同じものを用いた。佐藤らの手法及びＢＤＨで用いる基底は、人工データでは元の基底を分散の大きいものからν個選び、実データでは主成分分析で得られた主成分を分散の大きい方からν個を選んだ。

[0076] 図５、図６および表１に、ＡＮＮ、ＳＨ、佐藤らの手法およびこの発明に係るＢＤＨにおいて、最適なパラメータにおける精度（真の最近傍点が得られた割合）と処理時間（クエリを与えてから解を得るまでの時間の平均）の関係と、そのときのメモリ使用量を示す。ここでの最適とは同一精度で比較したときに処理時間が最も小さくなる状態を指す。予備実験の結果、パラメータとしてＳＨはビット長が$\log_2 n$であるとき、佐藤らの手法、ＢＤＨでは次元数ν＝$\log_2 n \times M$、分割幅Ｗ＝｛max（Ψν・ｐ）－min（Ψν・ｐ）｝／２であるときが最適であることがわかっている。これらのパラメータはハッシュサイズがデータ数ｎと同程度になる値である。

[0077] データは64次元、128次元、256次元の正規分布に従う人工データ（各基底で分散は100～400で一様に選ばれる）と、TRECVID2010のInstance Searchタスクで配布された動画の各フレーム画像から抽出したSIFT特徴量（128次元、なお、SIFTについては、例えばD.G. Lowe, "Distinctive image features from scale-invariant keypoints," International journal of computer vision, vol.60, no.2, pp.91-110, 2004.参照）の４種類をそれぞれ1000万点用意した。クエリはデータベースと同じ条件でつくられた2000点を用い、その平均を結果とする。

[0078] 人工データの結果については、精度と処理時間の関係を図５に示す。図５（ａ）は64次元の人工データ、図５（ｂ）は128次元の人工データ、図５（ｃ）は256次元の人工データの実験結果である。このときのメモリ使用量を表１に示す。画像データの結果については、図６に示す。なお、図５および図６では横軸を精度、縦軸を処理時間としている。凡例の単一ハッシュはこの発明においてハッシュテーブルを分割しなかった場合であり、分割ハッシュはハッシュテーブルを分割した場合である。

[0079] 以上の実験の結果、全てのデータにおいて同一精度で比較したときにＢＤＨが最も高速であった。人工データにおいて、単一ハッシュと分割ハッシュを比べると低次元のデータに対しては単一ハッシュの方がわずかに良い結果が得られているが、次元数が大きくなると、分割ハッシュの有効性が現れる。これは、ハッシュテーブル分割により探索の考慮に入る基底の数が増え、これによって推定距離の精度の低下を抑えることができたからである。故に、高次元データに対してハッシュテーブル分割が有効であるといえる。

SIFT特徴量（128次元）に対する結果を見ると、単一ハッシュが優勢である。処理時間を見ると、同精度で比べたときに64次元の人工データよりも高速に解を得られていることが分かる。つまり、SIFT特徴量は見かけ上128次元であるが、実質的な次元数は半分以下であり、それ故に単一ハッシュが優勢になったと考えられる。

[0080] [表1]

	64 次元	128 次元	256 次元
ANN	3.4GB	5.8GB	11GB
SH	3.0GB	5.3GB	10GB
佐藤らの手法	3.0GB	5.3GB	10GB
提案した BDH	3.0GB	5.3GB	10GB

以下では更なる三つのアプローチによってＢＤＨの改良を行う。一つ目は距離推定に適した空間分割の提案、二つ目は確率密度関数に基づく距離推定の提案、三つ目がクエリ周辺のデータ密度を考慮した探索領域の拡張である

。以降その詳細について述べる。

[0081] 〈第３の改良点：距離推定に適したデータ空間の分割〉

ＢＤＨはデータの一様分布を仮定して各基底を等分割し、その領域の中心をバケットに属する点の代表ベクトルとしたが、一般に実データは一様でなく、このような方法では距離推定の誤差が大きい。そこで発明者らは、図７のように各基底の要素の代表値をk-means法によって求め、データの分布に合わせた適応的な分割を施すことで推定誤差を最小化する。このアプローチにより、距離推定の精度が向上し、より効率的に最近傍候補を得ることができる。この発明では、第ｉ基底をk_i分割することを考え、各基底に適切な代表値をそれぞれk_i個用意し、誤差Ｅを最小化する。第ｉ基底のｊ番目の代表値をC_{ij}とおく。ハッシュ関数は次式で表される。

[0082] [数15]

$$H(\boldsymbol{x}) = \{h_1(\boldsymbol{x}), h_2(\boldsymbol{x}), \ldots, h_v(\boldsymbol{x})\} \tag{12}$$

$$h_i(\boldsymbol{x}) = \underset{1 \le j \le k_i}{\arg\min} |\boldsymbol{V}_i \cdot \boldsymbol{x} - C_{ij}| \tag{13}$$

ただし、V_iは第ｉ基底の方向の単位ベクトルである。

すると基底の代表値C_{ij}の組み合わせによりベクトルを表すことができるので、このベクトルによってバケットが定義されるものとする。すなわち、あるベクトルが最近傍となる領域をバケットの範囲と定める。このベクトルのことを代表ベクトルと呼ぶことにする。このようにしてデータ空間は$\Pi_i{}^v k_i$個の領域に分割される。従って、点ｐと代表値C_{ij}との第ｉ基底方向における距離成分、即ち基底距離BD_iは

[0083] [数16]

$$\mathrm{BD}_i(\boldsymbol{p}, j) = \{\boldsymbol{V}_i \cdot \boldsymbol{p} - C_{ij}\}^2 \tag{14}$$

のように表される。また、この場合の各サンプル（ベクトルデータ）と代表ベクトルの誤差は式（１５）で定義される。

[0084]

[数17]

$$E(\boldsymbol{X}) = \sum_{i=1}^{d} BE_i(\boldsymbol{X}) \tag{15}$$

$$BE_i(\boldsymbol{X}) = \frac{1}{N} \sum_{n=1}^{N} \mathrm{BD}_i(\boldsymbol{X}_n, h_i(\boldsymbol{X}_n)) \tag{16}$$

[0085] ただし、Xはデータの集合、X_nはn番目のデータを、BE_iは第i主成分方向の誤差を表している。この発明では、各基底における代表値$\{C_{ij}\}$をこの発明と同じ目的関数を持つk-means法によって求める。

ここで重要となるのは、各基底における代表値の数をいくらに設定するかということである。このような問題に対してはよく全基底を同数で分割するが、実際のデータにおいては主成分基底に射影されたデータの分散は主成分基底毎に大きく異なり、等価に扱うことは効率的でない。当然のことながら同じ代表ベクトル数ならば推定誤差が小さい方がよい。

[0086] ここで、ある基底の代表値の数をn倍する場合を考えると、代表ベクトル数もn倍となる。一様分布を仮定すると、量子化の影響からこのときの推定誤差は$1/n^2$倍となる。

分散の大きい基底ほど代表ベクトルの増加によって誤差が大きく減少すると期待されるため、分散の大きい基底から順に代表値を増やし、全ての基底の誤差が与えられた閾値Mより小さくなった時点で分割を終了することが効率的と考えられる。これは各基底の距離計算への寄与率を考慮した適応的な分割となっている。図7は、この実施形態に係る等分割と適応的な基底分割の比較例を示す説明図である。

[0087] 〈第4の改良点：確率密度関数に基づく距離推定〉

前節では一様分布の制約を緩和するために空間分割の構造を提案した。これに対して、本節では各区間の確率密度関数を仮定し、より一般の分布に対して柔軟に対応できる距離推定法を提案する。以下、手法の詳細を示す。

この発明では、各主成分方向に対して$h_i(p) = j$を満たす領域（$t_{i(j-1)} \leqq V_i \cdot p < t_{ij}$）にあるサンプル（ベクトルデータ）の分布をヒストグラムで

表し、これを正規化した後に最小二乗法によって確率密度関数$P_{ij}(y)$に変換する。ただし、t_{ij}は次のように表される。

[0088] [数18]

$$t_{ij} = \begin{cases} \min_n \boldsymbol{V}_i \cdot \boldsymbol{X}_n & \text{if } j = 0 \\ \frac{C_{ij}+C_{i(j+1)}}{2} & \text{if } j = 1, \ldots, k_i - 1 \\ \max_n \boldsymbol{V}_i \cdot \boldsymbol{X}_n & \text{if } j = k_i \end{cases} \quad (17)$$

すると、第i主成分方向の推定距離$BD_i(p, j)$は二乗基底距離の期待値として次のように表される。

[0089] [数19]

$$\mathrm{BD}_i(\boldsymbol{p}, j) = \int_{t_{i(j-1)}}^{t_{ij}} P_{ij}(y)(\boldsymbol{V}_i \cdot \boldsymbol{p} - y)^2 dy \quad (18)$$

ここで提案する距離推定の特徴はBDHの距離推定に比べて一般の複雑な分布への適応度が高いと同時に、計算コストが小さく高速な処理に適していることが挙げられる。

[0090] 〈第5の改良点：クエリ周辺のデータ密度を考慮した探索半径の拡張〉

ハッシュを用いる手法では一般にパラメータとして与えられた探索半径Rに従って探索領域を決定する。しかし、公知の文献（例えば、W. Dong, Z. Wang, W. Josephson, M. Charikar, and K. Li, "Modeling LSH for performance tuning," Proceeding of the 17th ACM conference on Information and knowledge management, pp.669-678 2008.参照）で述べられているように、近似最近傍探索において真の最近傍点が得られる確率や処理時間はパラメータによって大きく左右される。従って、全てのクエリを共通のRで処理する場合、十分な精度を得るためにはクエリが疎な領域にある場合も最近傍点が得られるようにある程度大きなRを与える必要がある。その結果としてクエリが密な領域にある場合には探索領域が必要以上に広く、距離計算が過剰になるという現象が起こる。つまり、データ空間内のクエリ位置によって望ましいRは大きく異なる。

[0091] そこで発明者らは探索半径Rではなく最近傍候補数cをパラメータとすることで、探索半径を適応的に変化させ、クエリに依らず処理時間を安定して

軽減する方法を提案する。これは、推定距離の小さい所から段階的に探索領域を拡張していき、最近傍候補数ｃを満たした時点で探索領域拡張を打ち切るというものである。図８は、この実施形態の例として、クエリ周辺の密度に応じて探索半径を変化させる様子を示す説明図である。なお、ハッシュの次元数ν＝２の例である。この手法により、図８（ａ）のようにクエリ周辺が疎な場合には広い範囲を、図８（ｂ）のようにクエリ周辺が密な場合は狭い範囲を探索することができる。密な領域に入ったクエリに対する過剰な距離計算を抑えることができるために結果として平均パフォーマンスが向上する。なお、図８（ａ）（ｂ）で灰色の領域は参照バケットを示しており、共に９点を最近傍候補としている。

[0092] 次に、この実施形態の手法に係る、効率的な近傍バケットの参照アルゴリズムについて述べる。従来の手法では、全てのバケットに対してハッシュ上の距離を計算して参照バケットを求めるもの（例えば、H. Jegou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.33, no.1, pp.117-128, 2011.参照）や、実装上ハッシュ構造を持たずに全サンプル点に対してハッシュ上の距離を計算することで探索半径以下の点を割り出すもの（ＳＨやその改良手法）など、最近傍候補の特定にも処理時間を要していた。

[0093] そこで各バケットを推定距離の小さい所から厳密に参照するという条件を緩和する。つまり、推定距離の上限を段階的に大きくし、上限以下のバケットの順位を無視して参照する。すると、複雑なデータ構造やソートといった処理が必要なくなり、高速に探索バケットを特定することができる。図１９は、この実施形態において探索半径を適応的に変化させる２つのアルゴリズムを示している。「アルゴリズム３」および「アルゴリズム４」である。ただし、これらのアルゴリズムは何れもクエリをｑとして、BD_{ij}＝BD_i（ｑ，ｊ）である。Ｌ，Ｕは、探索半径の下限および上限をそれぞれ表し、Δは探索半径を拡張するときの前回との半径の差分である。このような最近傍候補

数を基準とした段階的な処理により、前述の従来手法や「アルゴリズム１」、「アルゴリズム２」よりも効率的な探索が可能になる。

[0094] 「アルゴリズム１」および「アルゴリズム２」では、$\nu-i$個の各ハッシュ関数において取りうる最小値を足して得られる距離mD_iを算出してクエリから半径Ｒ以内に存在するバケットを探索したが、「アルゴリズム３」および「アルゴリズム４」ではそれに加えて、取りうる最大値を足して得られる距離MD_iを算出してクエリから半径Ｌ以遠に存在するという条件も使用する。

アルゴリズム３では１～４行目で、$\nu-i$次元分の距離の最小値mD_iと取りうる最大値を足して得られる距離ＭＤｉを算出する。５～６行目では探索半径の下限および上限であるＬとＵの初期値を設定する。７～１１行目では最近傍候補がc点以上になるまで探索半径をΔずつ拡張しながらアルゴリズム４の関数を呼び出して探索を続ける。

[0095] アルゴリズム４の２つの引数はアルゴリズム２と同じである。１～７行目では、最後の基底(ν番目のハッシュ値を決める段階)に到達していなければ（１行目），k_i個(第i基底の分割数)のハッシュ値を試す（2 行目）。BD_{ij}はハッシュ値jを選択した時の第i基底における1次元分の距離であるので、Ｄ＋BD_{ij}＋mD_iはi個のハッシュ値が決まった段階で最も近いバケットまでの距離であり、Ｄ＋BD_{ij}＋MD_iは最も遠いバケットまでの距離である。前者が探索半径の上限Ｕよりも小さく、かつ後者が探索半径の下限Ｌよりも大きい場合のみ自分自身を再帰呼び出しして次の基底に進む（３～５行目）。９～１３行目では、最後であるν番目の基底に到達したときのみ実行され，探索半径の上限以内かつ下限以遠のバケットがあればそのハッシュを引く。

[0096] 〈第３～５の改良点に係る実験〉

前述の第３～５の改良点について、その有効性を確認する実験を行った。以下に実験の内容とその結果を記す。

実験において、インプリメントは全てC++で行った。ＡＮＮはANN Library（URL:http://www.cs.umd.edu/ mount/ANN/参照）を用い、ＳＨは著者のMATLABのソースコード（URL: http://www.cs.huji.ac.il/yweiss/SpectralHashin

g/参照）とLSH-KIT（URL: http://lshkit.sourceforge.net/参照）を参考にして発明者らが実装した。実験に用いた計算機は、CPUがOpteron（tm）6174（2.2GHz）、メモリは256[GB]であり、実験はシングル・コアで行った。データベースにはTRECVID2010のInstance Searchタスクで配布された動画から１０秒おきに得た画像よりSIFT特徴量を抽出し、重複するベクトルを取り除いた。

[0097] 〈１．実験１〉

符号長を変化させたときの推定距離と実際のユークリッド距離の相関係数を示す。この結果は、推定距離がどの程度真の距離を反映しているかを表す。比較するハッシュ構造はＳＨ、ＢＤＨ、k-means ＢＤＨ（第３の改良点：距離推定に適したデータ空間の分割）であり、SIFT特徴量100万点に対して1000点のクエリを入力し、相関係数の平均を結果とした。この実験においてＳＨの推定距離はバイナリ符号のハミング距離であり、ＢＤＨ、k-means ＢＤＨの符号長はｂをバケット数（ハッシュサイズ）として［$\log_2$（ｂ）］である。推定距離の精度を示す結果を図９に示す。横軸が符号長［bit］、縦軸が相関係数である。k-means法による分布を考慮した分割により、同じ符号長で比較してＢＤＨよりも高い相関係数が得られている。符号長を大きくしてもＳＨの相関係数が大きくならないのは、ＳＨが一様分布を仮定した空間分割をしていることや、ハミング距離とユークリッド距離の距離尺度としてのずれが原因である。

参考までに符号長を120bitとしたときの推定距離と真の距離の関係を図１０に示す。やはり、相関係数の結果が示すようにk-means ＢＤＨの推定距離が真の距離よく反映しているのがわかる。

[0098] 〈２．実験２〉

符号長を変化させたときのハッシュ上の距離における最近傍点の平均順位を示す。つまりこれは最近傍点を発見するために最低限必要な計算量を示していて、この値が小さいほど良い。この実験はk-近傍探索における符号長-Precision曲線による評価に相当する。ただし、ＢＤＨやk-means ＢＤＨの符号

長はｃをバケット数（ハッシュサイズ）として［$\log_2$（ｃ）］と定義した。また、SIFT特徴量100万点に対して1000点のクエリを入力し、その平均を結果とした。符号長を変化させたときのハッシュ上の距離における最近傍点の平均順位を図１１に示す。

横軸が符号長［bit］、縦軸が最近傍点の順位である。どの符号長においてもＢＤＨがＳＨに比べて最近傍点の平均順位が高くなっていることがわかる。最近傍点の順位はＳＨに比べて20bitで約1/8、40bitで約1/8、80bitで約1/13となった。

［0099］ 〈３．実験３〉

探索パラメータを変化させたときの、最近傍探索問題における精度（真の最近傍点が得られた割合）と処理時間（クエリを与えてから解を得るまでの時間の平均）の関係を示す。本実験ではデータベースとしてSIFT特徴量1000万点、クエリとしてデータベースと同じ条件でつくられた1000点を用いた。ＳＨ及びＢＤＨの符号長ｎは24［bit］とした。ｎ＝24としたのは実験的にハッシュサイズ2^{24}データ数10^7となるときが最適であると分かっている為である。

［0100］ まず初めに、探索半径Ｒを固定した場合のデータ分布に適応した距離推定の有効性を確認する。比較の結果を図１２に示す。k-means ＢＤＨは第３の改良点に係るk-meansを用いた空間分割法を適用したもの、k-means ＢＤＨＰはk-means ＢＤＨに第４の改良点である確率密度関数を用いた距離推定を適用したものである。k-meansのみを適応すると、ＢＤＨよりも性能が落ちるが、確率密度関数を組み合わせることで改善が見られた。

［0101］ 次に、図１２で最も高速であった第４の改良点（確率密度関数に基づく距離推定）のみのk-means ＢＤＨ Ｐ（探索半径Ｒを固定する方法）と、第４の改良点に第５の改良点（最近傍候補数ｃを固定する方法）を加えたk-means ＢＤＨ ＰＣの比較を行う。比較の結果を図１３に示す。クエリ周辺のデータ密度を反映させることにより、概ね２倍程度の高速化が確認され、十分な効果が得られることが分かった。

[0102] 最後に代表的既存手法であるＡＮＮとＳＨとの比較を行う。また、それぞれの探索パラメータはＡＮＮがε、ＳＨがハミング距離Ｒ、k-means ＢＤＨ ＰＣ最近傍候補数cである。比較結果を図１４に示す。k-means ＢＤＨ ＰＣの処理時間は同一精度で比較して、ＡＮＮに対して約1/10、ＳＨに対して約1/6～1/12となっており、大幅な高速化がなされていることが分かる。

[0103] 〈４．実験４〉

次にこの発明のパラメータチューニングに関する実験を行う。探索を実行する際に、入力するパラメータとそのときに得られる性能の関係を知ることは実用上非常に重要であり、この発明は容易にこの関係を知ることができる。入力した最近傍候補数のパラメータｃと処理時間、精度の関係をそれぞれ図１５、１６に示す。図１５から処理時間はパラメータｃに対してほぼ線形であり、処理時間Ｔ＝ｔｃで表すことができる。また、図１６は縦横軸をlogスケールで示しており、精度20%～90%の範囲でグラフがほぼ直線を描いており、この範囲において精度Ａ＝c^aが成り立つことを示している。20%以下の範囲でグラフが直線から離れるのは、図１７に示されるようにｃが小さいとほとんどのクエリは一回目の探索でｃ個の最近傍候補を確保する為に、実際に得られる最近傍候補数が変わらないためである。従って、ある程度のサンプルクエリを入力しておけば、パラメータｃから精度と処理時間を予め知ることができ、パラメータチューニングが容易である。

[0104] 前述した実施の形態の他にも、この発明について種々の変形例があり得る。それらの変形例は、この発明の範囲に属さないと解されるべきものではない。この発明には、請求の範囲と均等の意味および前記範囲内でのすべての変形とが含まれるべきである。

例えば、この発明は、距離尺度にマンハッタン距離（L_1ノルム）を用いた場合にも、あるいはユークリット距離（L_2ノルム）を用いた場合にも、また、それ以外の距離を用いた場合にも適用可能である。

[0105] 〈第６の改良点〉

第３の改良点の「距離推定に適したデータ空間の分割」に係るインデクシ

ングと第４の改良点の「確率密度関数に基づく距離推定」を発展させた第６の改良点を述べる。この方法を第５の改良点の「クエリ周辺のデータ密度を考慮した探索半径の拡張」に係る探索方式と組み合わせて使用することで高い探索性能を実現できることを実験で確認する。

[0106] １．データ空間のインデクシング

第３の改良ではデータ空間を分割（インデクシング）する際に、選択した基底によって張られるＭ次元空間をスカラ量子化によって粗量子化している。しかし、実データにおいては基底が直交していてもそれらが必ずしも互いに独立とはいえず、スカラ量子化では量子化効率が悪い。

[0107] ここでハッシュ関数をｈ^m（ｘ）、データが存在する範囲を覆うのに十分なハッシュ値の個数をｇ^{m}とすることで、バケット数Ｇは

[数20]

$$G = \prod_m g^m \tag{19}$$

$$g^m = \max h^m(\boldsymbol{x}) - \min h^m(\boldsymbol{x}) + 1 \tag{20}$$

[0108] のようにＭに対して指数的に大きくなる。バケットの数が大きくなれば、精度は上がるが隣接バケットを探索する処理に時間がかかるようになる。従って、安易に大きなＭを用いることはできず、ベクトルが高次元であった場合には全次元数に対してインデクシングに使うＭが相対的に小さくなり、距離推定の精度が得られずに近似最近傍探索全体の性能の低下を招く。経験的にバケット数がデータセットサイズ

[0109] [数21]

$$|\boldsymbol{X}|$$

に等しくなる程度が最もバランスが良いことが分かっている。

[0110] そこで、第６の改良点では直交基底Ｖの中から、Ｐ本の直交基底からなるセットをＭ組選択し、データ空間をＭ個のＰ次元部分空間の直積で表現する。ここで、Ｖ^{m}をｍ番目の部分空間を張るＰ個の直交基底とする。インデクシ

ングの際には部分空間毎にk-meansアルゴリズムを用いてセントロイドの集合C^mを求め、量子化誤差の最小化を図る。これは上記の基底によって張られたＰＭ次元空間に射影されたベクトルを、Ｍ個のＰ次元部分ベクトルに分けてプロダクト量子化することに等しい。C^mのｉ番目の部分セントロイドをC_i^mと表せば、ハッシュ関数Ｈ（・）は次のようになる。

[0111] [数22]

$$H(\boldsymbol{x}) = \{h^1(\boldsymbol{x}), h^2(\boldsymbol{x}), \ldots, h^M(\boldsymbol{x})\} \quad (21)$$

$$h^m(\boldsymbol{x}) = \arg\min_i D(\boldsymbol{c}_i^m, \boldsymbol{V}^m\boldsymbol{x}) \quad (22)$$

[0112] ２．バケット距離

ここでは推定誤差が最小となるような距離の推定量を導き、バケット距離を定義する。各基底間に相関がないと仮定（主成分基底を用いれば２次までの無相関が保障される）すれば、各基底方向の推定距離の誤差を独立に最小化すればよいので、１次元の二乗距離の誤差を最小化することを考える。

最小化問題を次のように定義する。データｘは確率密度関数Ｐ（ｘ）に従うものとし、ｘがある領域ｚに存在する事象をＺとすれば、領域ｚ内にあるデータの分布はＰ（ｘ|Ｚ）と表される。このとき、

[0113] [数23]

$$\tilde{x} = \{x \in z\}$$

と置くことで

[数24]

$$P(\tilde{x}) = P(x|Z)$$

となる。このような条件の下でクエリｑと

[0114] [数25]

$$\tilde{x}$$

の二乗距離

[数26]

$$D^2(q,\tilde{x})$$

の推定量をｅとすると、誤差の最小化問題は次のようにおける。

[0115] [数27]

$$\min \mathrm{E}_{\tilde{x}}[\{D^2(q,\tilde{x})-e\}^2] \tag{23}$$

この式の形は分散の定義式と同じであるため、式（２３）を最小化するｅは以下のように

[数28]

$$D^2(q,\tilde{x})$$

の期待値として得られる。

[0116] [数29]

$$\begin{aligned} e =& \mathrm{E}_{\tilde{x}}[D^2(q,\tilde{x})] \\ =& q^2 - 2\mathrm{E}[\tilde{x}]\,q + \mathrm{E}\left[\tilde{x}^2\right] \\ =& (q-\mathrm{E}[\tilde{x}])^2 + \mathrm{Var}[\tilde{x}] \end{aligned} \tag{24}$$

[0117] 式（２４）で、Var［］は、引数を母集団の標本とみなして不偏分散を返す関数である。ここで得られる推定量は領域の重心との距離ではないことに注意べきである。

以上の結果から、ハッシュ値リストがＨであるようなバケット距離F_Hを次のように定義する。なお、下記式（２６）で、u^m_pは第ｍ番目の部分空間の第ｐ番目の主成分基底である。

[数30]

$$F_H = \sum_{m=1}^{M} f^m_{h^m} \tag{25}$$

$$f^m_i = D^2(\boldsymbol{q},\boldsymbol{c}^m_i) + \sum_p \mathrm{Var}\left[\boldsymbol{u}^m_p \boldsymbol{x} | h^m(\boldsymbol{x}) = i\right] \tag{26}$$

[0118] ３．部分セントロイドの数

次に各部分空間に与えるセントロイドの数g_mを考える。一般的なプロダクト量子化では、g_mは各部分空間で共通の値を用いるが、ＰＣＡを用いた場合など、各部分空間のデータの広がりに偏りがある場合は量子化効率が悪く、図２４のように量子化誤差を最小化するにはg_mを各部分空間の広がりに合わせて調節する必要がある。具体的な説明をする。図２４はデータ空間中の点の分布の様子、u_1，u_2は第１、第２主成分基底を表す。図２４（ａ）のNormalのように、基の基底を独立に量子化すると、これらは相関を持っている場合が多く量子化に無駄が生じる。そこで図２４（ｂ）のＰＣＡのように、主成分方向に射影すれば、２次以下の相関を打ち消すことができる。しかし、この場合基底間の分散に偏りが生じるため分割される領域がある方向に長くなるような形になる。一般的にこのような領域分割は各領域が球に近づくことが理想であるため、これも誤差を大きくする原因となる。従って、図２４（ｃ）の“PCA+bit coordination”のように、点の分布を考慮して分割数を変動させればよい。

次に、g_mを分配する基準について考える。式（２３）の誤差を式（２４）の推定量ｅを用いて計算し直すと、次のようになる。

[0119] [数31]

$$\begin{aligned}
&\mathrm{E}_{\tilde{x},q}[\{D^2(q,\tilde{x})-e)\}^2] \\
=&4\mathrm{E}[q^2](\mathrm{E}[\tilde{x}^2]-\mathrm{E}[\tilde{x}]^2)+\mathrm{E}[\tilde{x}^4]-\mathrm{E}[\tilde{x}^2]^2 \\
&+4\mathrm{E}[q](\mathrm{E}[\tilde{x}]\mathrm{E}[\tilde{x}^2]-\mathrm{E}[\tilde{x}^3]) \\
=&4\mathrm{E}[q^2]\mathrm{Var}[\tilde{x}]+\mathrm{Var}[\tilde{x}^2]+4\mathrm{E}[q](\mathrm{E}[\tilde{x}]\mathrm{E}[\tilde{x}^2]-\mathrm{E}[\tilde{x}^3]) \\
\simeq&4\mathrm{E}[q^2]\mathrm{Var}[\tilde{x}] \qquad (27)
\end{aligned}$$

[0120] 最後の近似は

[数32]

$$\tilde{x}$$

は領域ｚ内に縛られているためにｑに対して分散が小さく、Ｅ［q^2］が他の要素に比べて非常に大きくなるため為された。従って空間インデクシングの際には、分割された領域内の分散

[数33]

$$\mathrm{Var}[\tilde{x}]$$

に注意すればよく、これは各部分空間の量子化誤差に相当する。以上から、ＢＤＨでは各部分空間の量子化誤差の最大値が最も小さくなるように、分散が大きい基底セットほどg^mを大きく設定する。

[0121] 〈第６の改良点に係る実験〉

ここでは、初めにメモリ使用量を制限せずに精度と処理時間の関係を検証し、次にメモリ削減を適用して、使用できるメモリ使用量に上限を設けた場合の精度と処理時間の関係を検証した。問題設定はクエリに近いＫ点を探索するＫ近傍探索問題とした。

[0122] １．実験条件

実験にはBIGANNデータセット（"Datasets for approximate nearest neighbor search," http://corpus-texmex.irisa.fr/.参照）のSIFT特徴量と80 million tiny images ("Tiny Images Dataset," http://groups.csail.mit.edu/vision/TinyImages/.参照）のGIST特徴量を用いた。クエリはともに1000個とし、探索精度は平均再現率、処理時間はクエリベクトル１個当たりの処理時間の平均である。SIFTの学習にはデータセットの初めの100万点、GISTの学習には10万点を用いた。実験に用いたＣＰＵはOpteron(tm)6174(2.2GHz)であり、実験は全てシングルコアで行った。

[0123] ２．処理時間と精度の関係

本節ではこの実施形態による提案手法と各比較手法の処理時間と精度の関係を示す。Ｋ＝１とした。ここではメモリ削減をしないため、どの手法も探索領域を広げることで必ず100%の精度を達成できる。実験にはSIFTが100万点(学習データ)、1000万点、１億点、GISTが10万点(学習データ)、100万点、1000万点のデータベースを用いた。比較手法は木構造を用いた手法としてRandamized kd-tree (C. Silpa-Anan and R. Hartley, "Optimised kd-trees for fast image descriptor matching," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.1-8, 2008.参照）、階層的k-m

eans (D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), vol.2, pp.2161-2168, 2006.参照）、ハッシュ法を用いたものとしてIVFADC, ＩＭＩを用いた。結果を図２５～図３０に示す。また、実験に用いたパラメータを表２に示す。図の縦軸は探索処理に要した時間を、横軸は精度をそれぞれ表している。グラフの凡例のうち"ＢＤＨ"はこの実施形態による提案手法を示している。これに対して、"Multi"はＩＭＩ、"IVF"はIVFADC、"RKD"はRandamized kd-tree、"HKM"は階層的k-meansの比較手法をそれぞれ示している。

[0124] [表2]

手法＼データセット	SIFT			GIST		
	1M	10M	100M	100K	1M	10M
BDH	20/5	26/3	28/5	18/3	22/5	24/5
IMI	14	18	20	16	18	22
IVFADC	10	12	14	8	12	14
Randamized kd-tree	8	8	8	16	8	16
階層的k-means	64	64		16	64	32

[0125] 図２５～図３０のグラフからわかるように、提案手法はいずれの特徴量で比較しても、常に既存手法に比べて高速に解を得られることが確認できた。精度が90%を超えるような高精度帯においては各手法間にそれほど大きな違いは見られないが、50%以下の低精度帯においては提案手法が明らかに他と比べて高速である。特に点数の多い図２７で顕著である。

この原因は最近候補選択のオーバーヘッドである。IVFADCは近傍セントロイドの探索にＧ回の距離計算を行い、ＩＭＩは部分距離リスト $\{f_i^m\}$ の生成に$2G^{1/2}$回の距離計算とそのソートを必要とする為、低精度であっても処理時間が小さくならない。しかし、空間インデクシングの量子化精度はＢＤＨに比べて高いので、精度に対するグラフの上昇は緩やかになる。

符号の説明

[0126] １１：特徴点抽出部、 １３：探索範囲決定部、 １５：画像データベ

ース、　　１５ｈ：ハッシュテーブル、　　１７：最近傍点決定部、　　１９：投票部、　　２１：投票テーブル、　　２３：画像選択部

請求の範囲

[請求項1] ベクトルデータで表現される複数の点が入力されたとき、多次元ハッシュテーブルのインデックス算出用のハッシュ関数をそれぞれ適用して各点についてハッシュインデックスを算出し、前記多次元ハッシュテーブルのビンによって複数の領域に分割された多次元空間内の前記ハッシュインデックスに応じた領域に各点を射影することにより各点を多次元ハッシュテーブルに格納してなるデータベース格納部と、

クエリが入力されたとき、そのクエリに前記ハッシュ関数を適用して前記多次元空間内でのクエリの位置を決定し、クエリと前記空間内の各領域との距離の推定値を決定し、その推定値に基づいて少なくとも１つの探索すべき領域を決定する探索範囲決定部と、

前記探索すべき領域内の各点とクエリとの距離を計算し、クエリに最も近い点をクエリの最近傍点として算出する最近傍点決定部とを備え、

前記探索範囲決定部は、各領域のインデックスを参照してその領域の代表点を求め、前記クエリと各代表点との距離に基づいて前記推定値を決定し、分枝限定法を適用して前記探索すべき領域になり得ない領域を除外して前記探索すべき領域を決定することを特徴とする近似最近傍探索装置。

[請求項2] 前記データベース格納部は、Ｍ組（Ｍは２以上の自然数）のハッシュ関数群を用いてＭ組の多次元ハッシュテーブルに各点をそれぞれ登録してなる請求項１に記載の近似最近傍探索装置。

[請求項3] Ｍ組の多次元ハッシュテーブルは、各組の次元数が略等しい請求項２に記載の近似最近傍探索装置。

[請求項4] 前記多次元ハッシュテーブルは、各次元に対応するハッシュテーブルを組み合わせてなり、各ハッシュテーブルは各次元に対応する基底をビンで分割し、各ビンの幅は、すべてのビンが等幅とした場合に各ビンに登録されることになる点と、各ビンを代表する代表点との位置

の誤差を各ビンについて求め、それらの誤差の和がより小さくなるように各ビンの幅が調整されてなる請求項１～３の何れか一つに記載の近似最近傍探索装置。

［請求項5］ 各ハッシュテーブルは、各点を予め定められたｎ個のクラスタにクラスタリングしてクラスタごとの代表点を算出し、それぞれのクラスタに属する各点からそのクラスタの代表点までの平均距離を表す分散が予め定められた閾値よりも小さくなるように各ビンの幅が決定されてなる請求項４に記載の近似最近傍探索装置。

［請求項6］ 前記探索範囲決定部は、各基底の方向におけるベクトルデータの分布から確率密度関数を求め、その確率密度関数を距離の重み付けに用いて前記推定値を決定する請求項４または５に記載の近似最近傍探索装置。

［請求項7］ 前記探索範囲決定部は、クエリを中心として予め定められた探索半径Ｒの範囲内に代表点がある領域を前記探索すべき領域とする請求項１～６の何れか一つに記載の近似最近傍探索装置。

［請求項8］ 前記探索範囲決定部は、クエリを中心として探索半径Ｒの範囲内に代表点のある領域を前記探索すべき領域とし、その領域に含まれる点の数が予め定められた数に達するまで探索半径Ｒを漸次大きくする請求項１～６の何れか一つに記載の近似最近傍探索装置。

［請求項9］ 前記データベース格納部は、ν次元空間に各点が射影され、
前記多次元ハッシュテーブルは、前記ν次元空間を張るν本の直交基底のうちＰ本の直交基底が張る部分空間をＭ組選択し、各部分空間にk-means法を用いてその部分空間が分割され、かつ、各領域内の分散が大きい部分空間ほど分割の数が大きく設定されてなり、
前記検索範囲決定部は、各基底の方向におけるベクトルデータの分布から求めた確率密度関数に従って各領域に存在する点とクエリとの二乗距離の推定誤差が各基底方向においてそれぞれ最小化されるように前記推定値を決定する請求項１～８の何れか一つに記載の近似最近傍

探索装置。

[請求項10] 前記データベース格納部は、主成分分析により基底が決定されたν次元空間に各点が射影されてなり、

前記探索範囲決定部は、前記クエリと各代表点との距離の各基底の方向における距離成分をそれぞれ算出して前記推定値とし、

前記分枝限定法は、各距離成分の算出の過程において、各距離成分の和が定められた探索半径Ｒ内であることを制約条件とし、前記主成分分析において大きな固有値を有する基底の順に各領域の代表点が半径Ｒ内にあるかを判断する請求項１～９の何れか一つに記載の近似最近傍探索装置。

[請求項11] コンピュータが、

ベクトルデータで表現される複数の点が入力されたとき、多次元ハッシュテーブルのインデックス算出用のハッシュ関数をそれぞれ適用して各点についてハッシュインデックスを算出し、前記多次元ハッシュテーブルのビンによって複数の領域に分割された多次元空間内の前記ハッシュインデックスに応じた領域に各点を射影することにより各点を多次元ハッシュテーブルに格納してなるデータベース格納部にアクセスするステップと、

クエリが入力されたとき、そのクエリに前記ハッシュ関数を適用して前記空間内でのクエリの位置を決定し、クエリと前記空間内の各領域との距離の推定値を決定し、その推定値に基づいて少なくとも１つの探索すべき領域を決定する探索範囲決定ステップと、

前記探索すべき領域内の各点とクエリとの距離を計算し、クエリに最も近い点をクエリの最近傍点として算出するステップとを備え、

前記探索範囲決定ステップは、各領域のインデックスを参照してその領域の代表点を求め、前記クエリと各代表点との距離に基づいて前記推定値を決定し、分枝限定法を適用して前記探索すべき領域になり得ない領域を除外して前記探索すべき領域を決定することを特徴とする

近似最近傍探索方法。

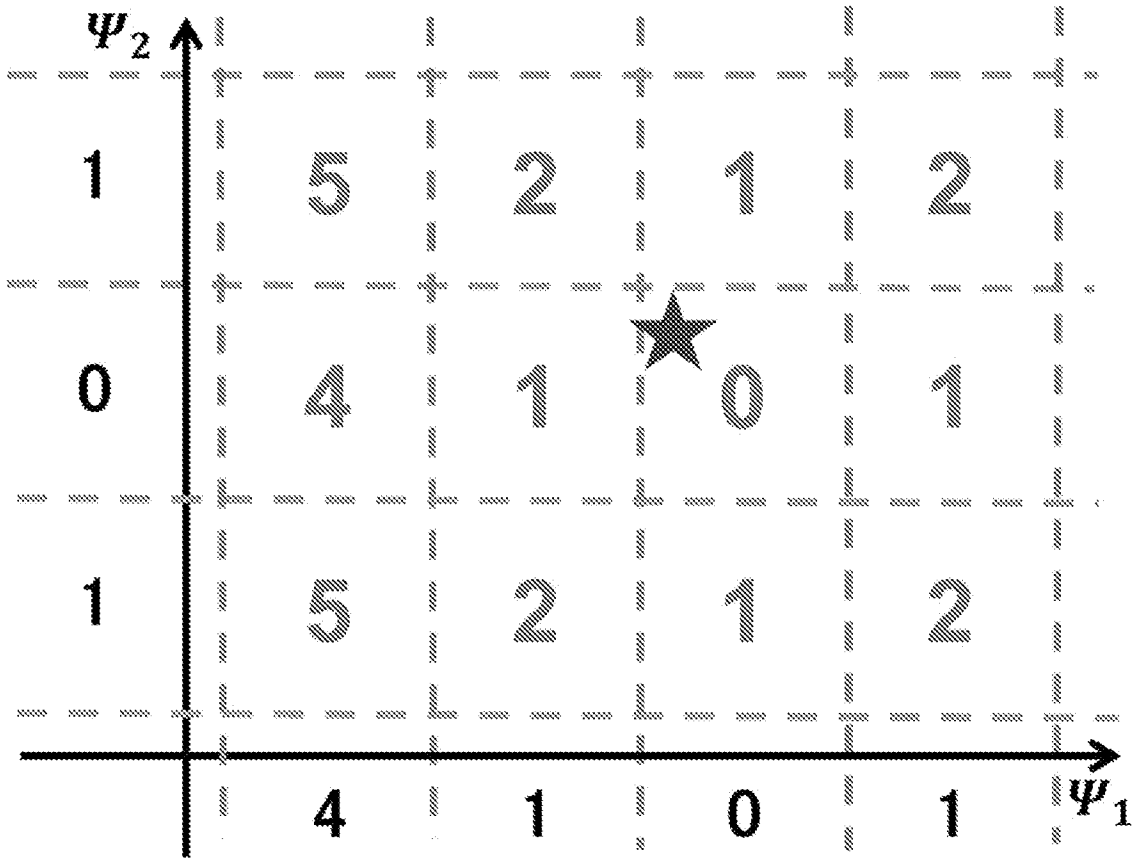
[請求項12] ベクトルデータで表現される複数の点が入力されたとき、多次元ハッシュテーブルのインデックス算出用のハッシュ関数をそれぞれ適用して各点についてハッシュインデックスを算出し、前記多次元ハッシュテーブルのビンによって複数の領域に分割された多次元空間内のハッシュインデックスに応じた領域に各点を射影することにより各点を多次元ハッシュテーブルに格納してなるデータベース格納部にアクセスする処理と、

クエリが入力されたとき、そのクエリに前記ハッシュ関数を適用して前記空間内でのクエリの位置を決定し、クエリと前記空間内の各領域との距離の推定値を決定し、その推定値に基づいて少なくとも１つの探索すべき領域を決定する探索範囲決定部としての処理と、

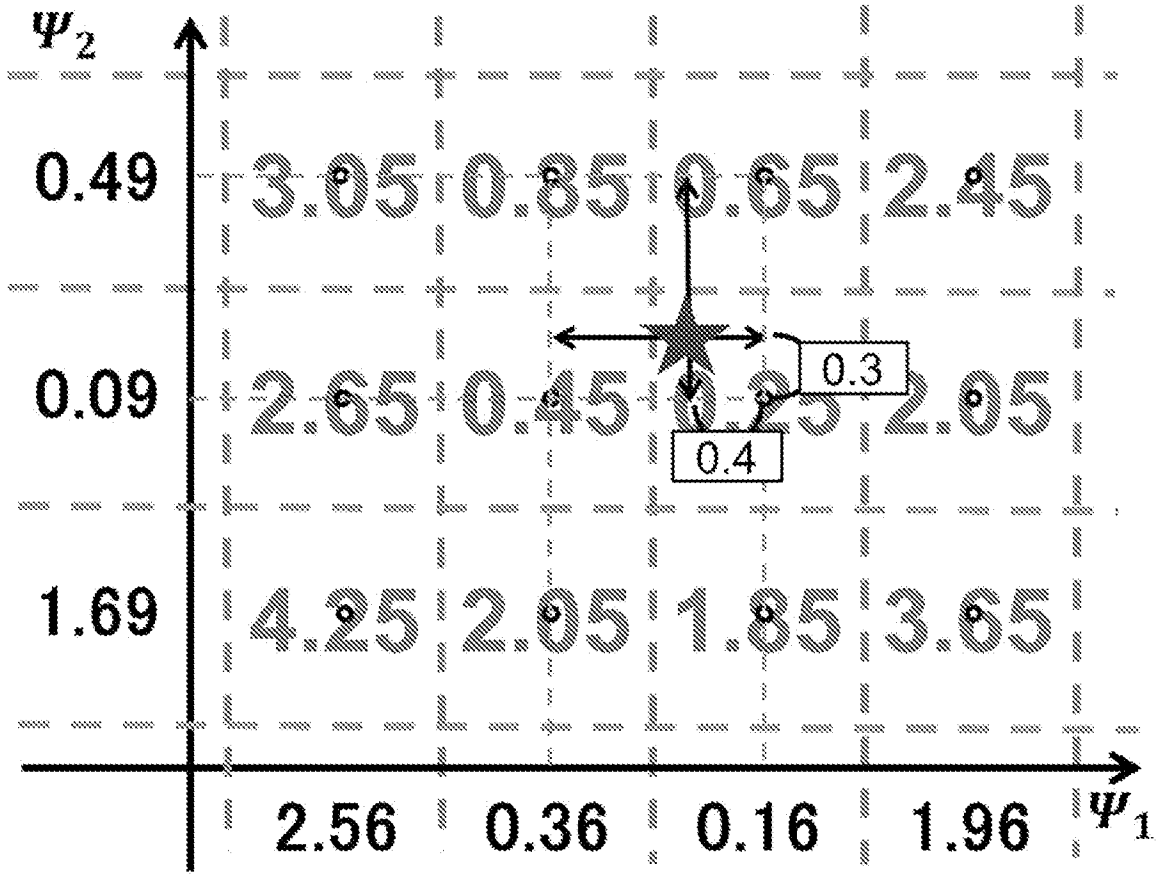
前記探索すべき領域内の各点とクエリとの距離を計算し、クエリに最も近い点をクエリの最近傍点として算出する最近傍点決定部としての処理をコンピュータに実行させ、

前記探索範囲決定部は、各領域のインデックスを参照してその領域の代表点を求め、前記クエリと各代表点との距離に基づいて前記推定値を決定し、分枝限定法を適用して前記探索すべき領域になり得ない領域を除外して前記探索すべき領域を決定することを特徴とする近似最近傍探索プログラム。

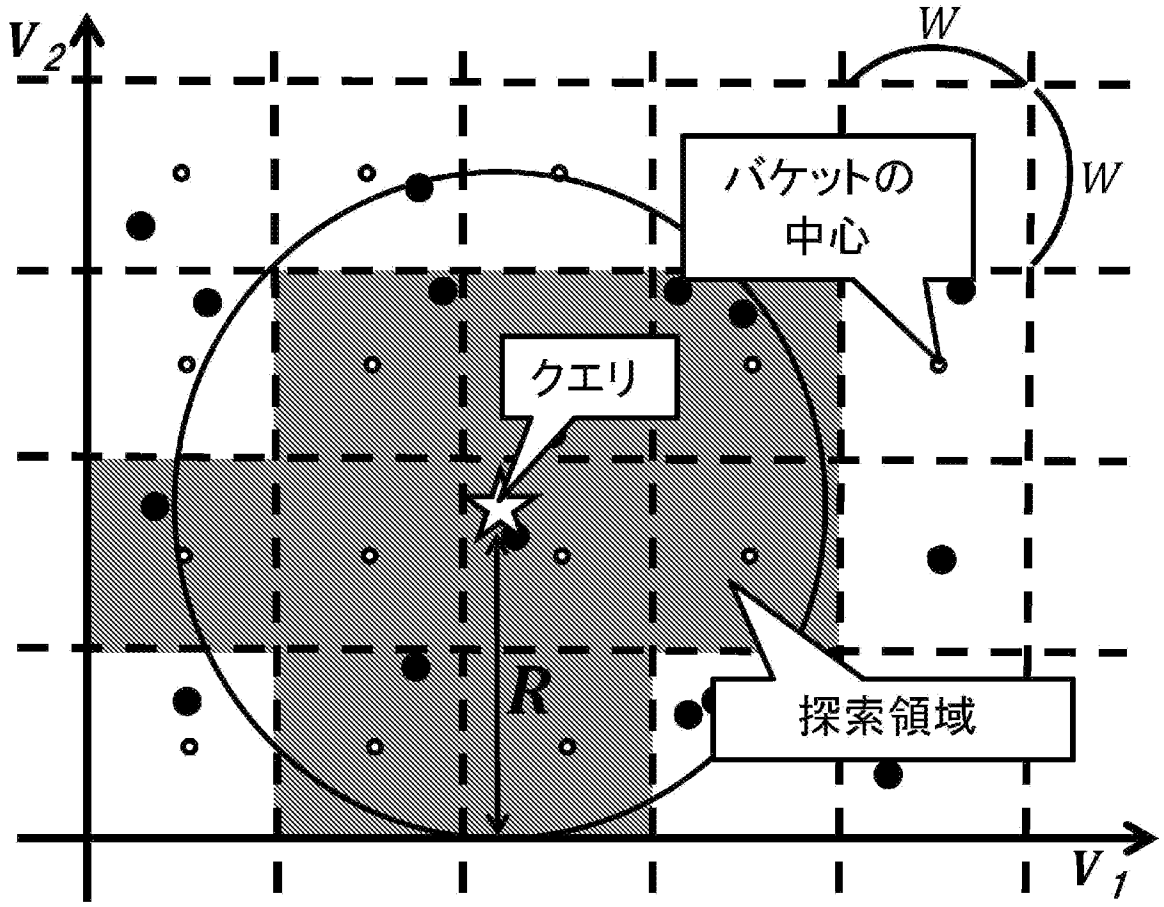
[図1]



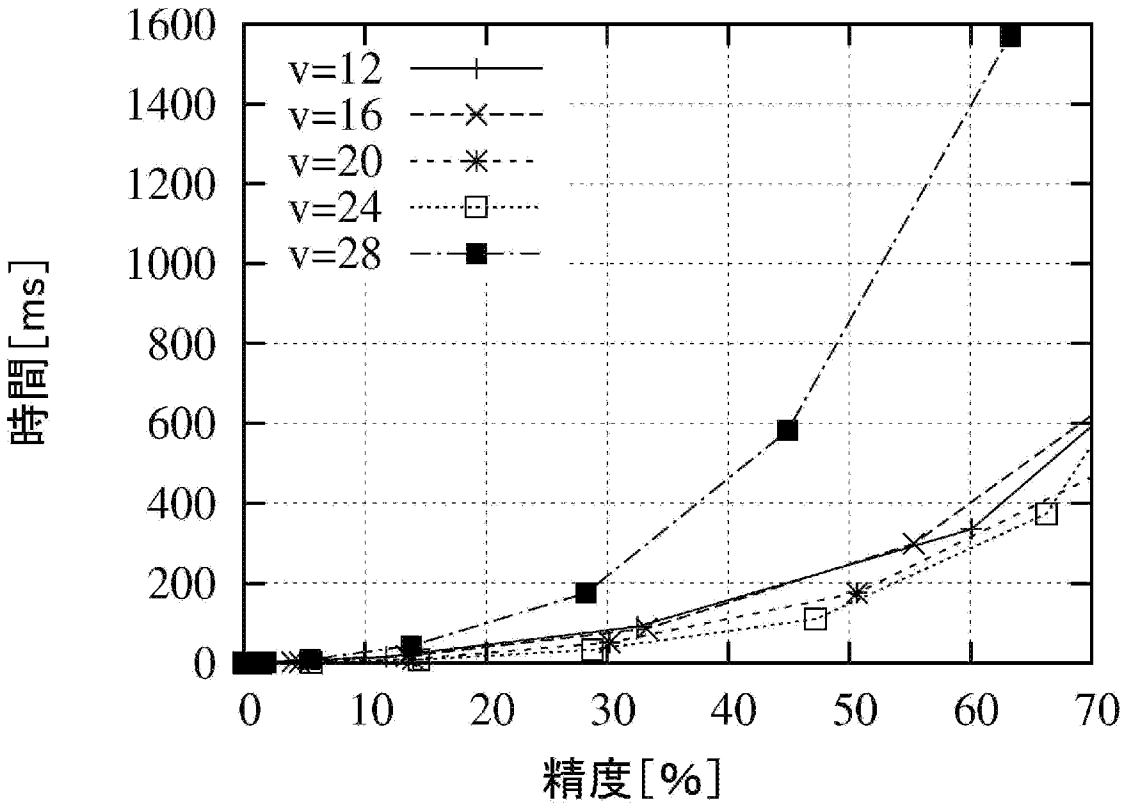
[図2]


Ψ2
0.49
3.05
0.85
0.65
2.45
0.09
2.65
0.45
0.25
2.05
1.69
4.25
2.05
1.85
3.65
2.56
0.36
0.16
1.96
Ψ1
0.3
0.4

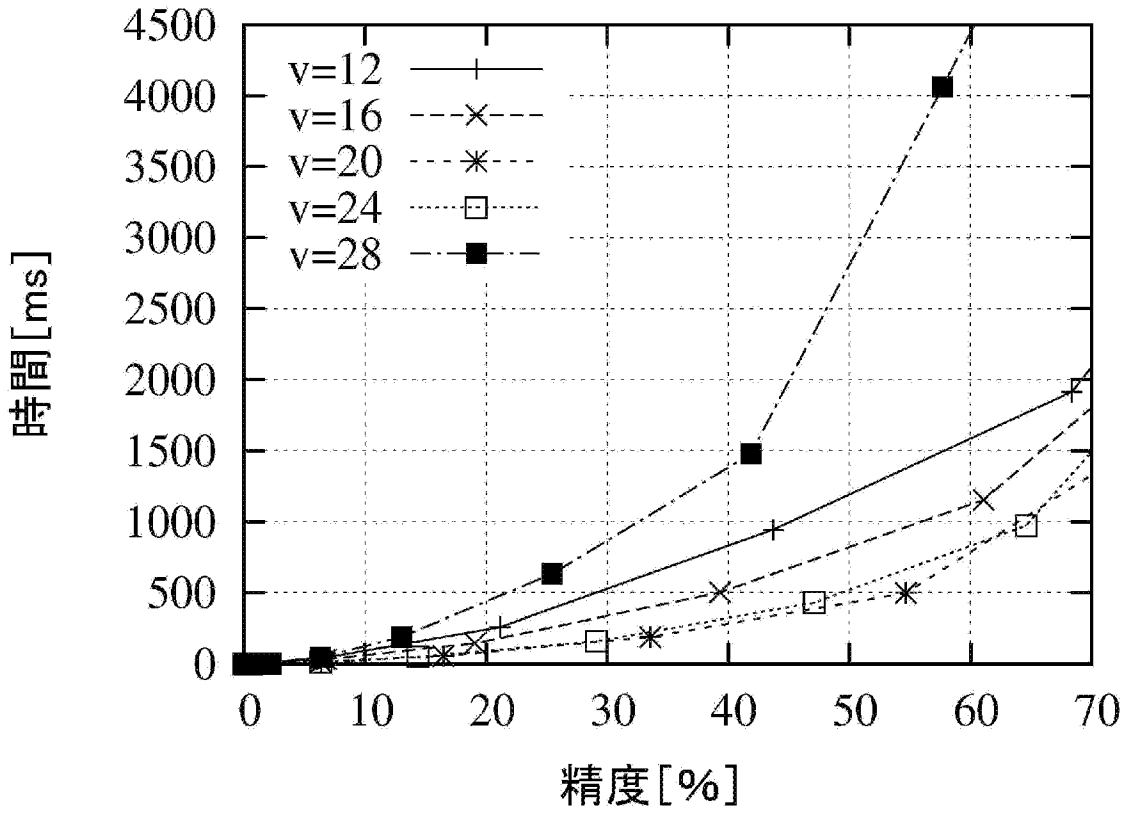
[図3]



[図4]

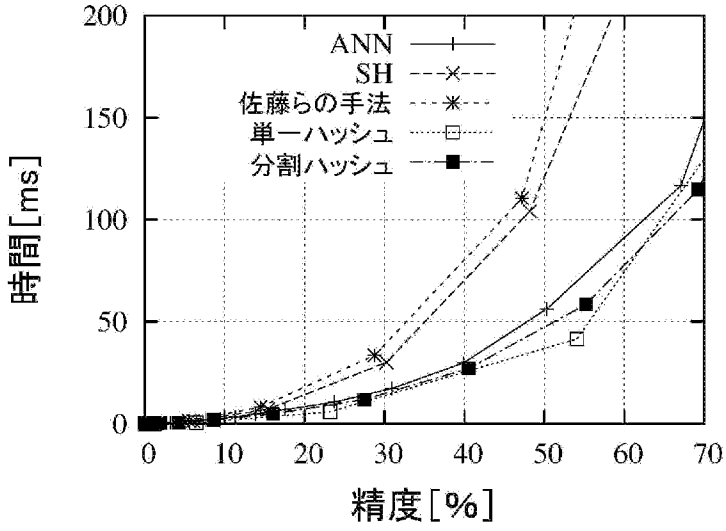


(a)64次元

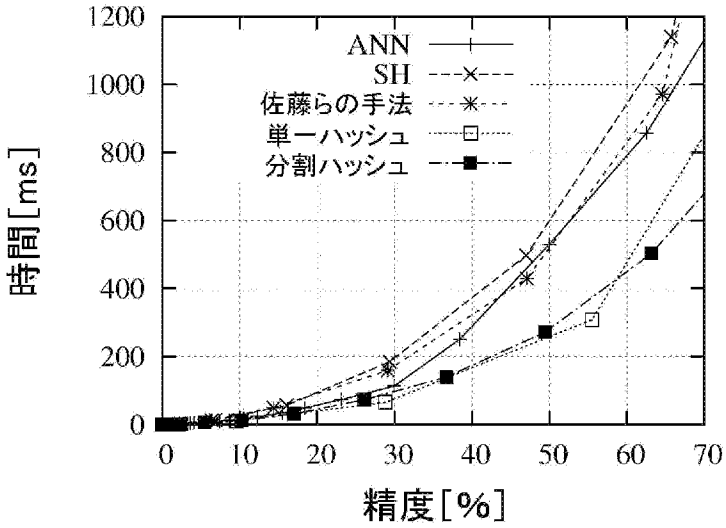


(b)128次元

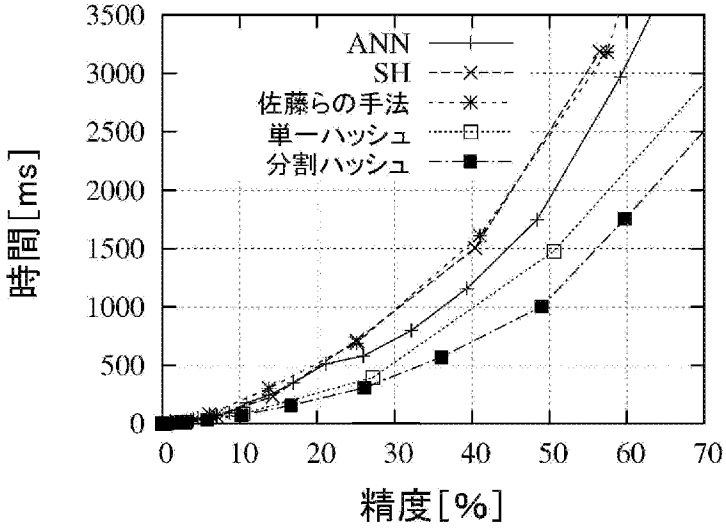
[図5]



(a) 64 次元 1000 万点

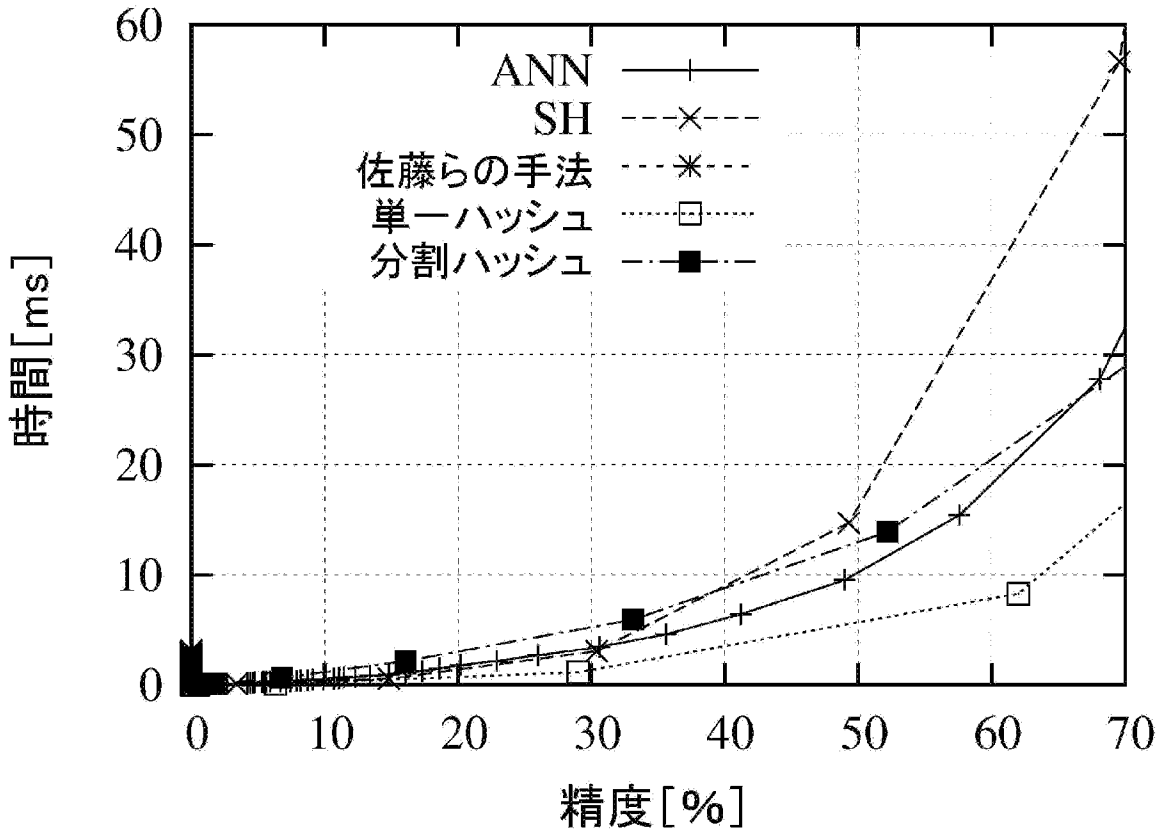


(b) 128 次元 1000 万点

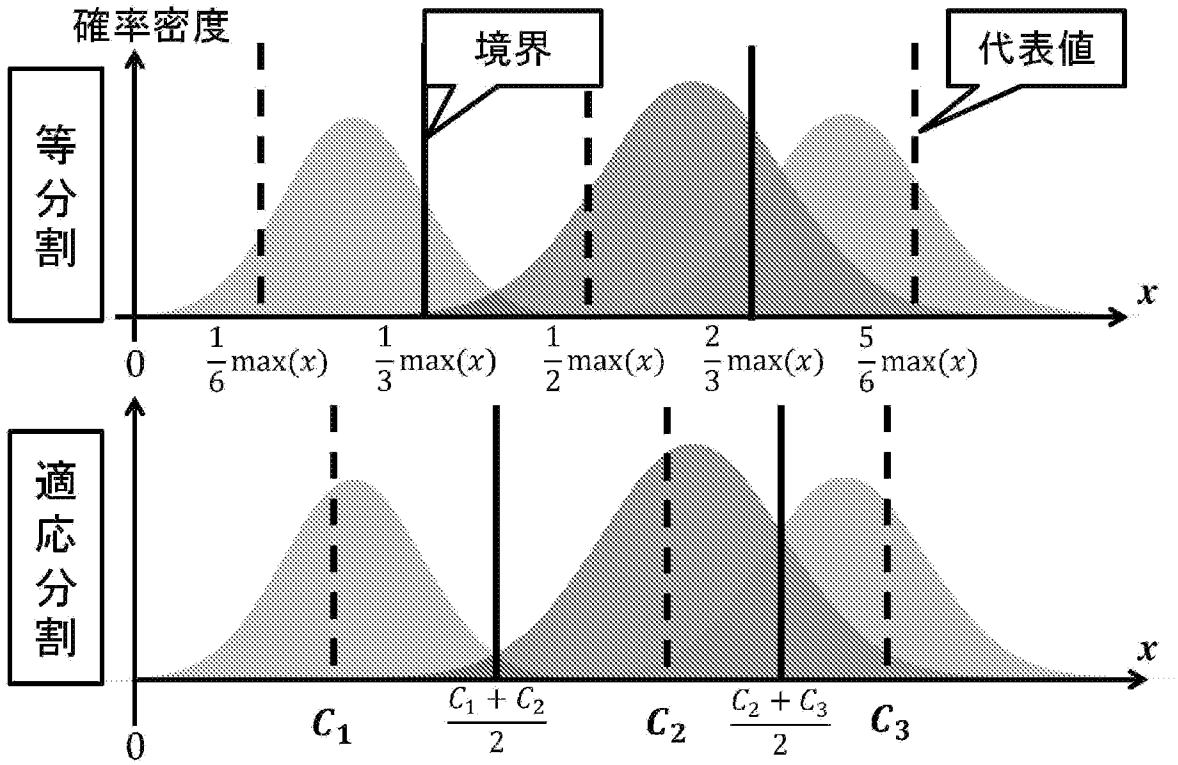


(c) 256 次元 1000 万点

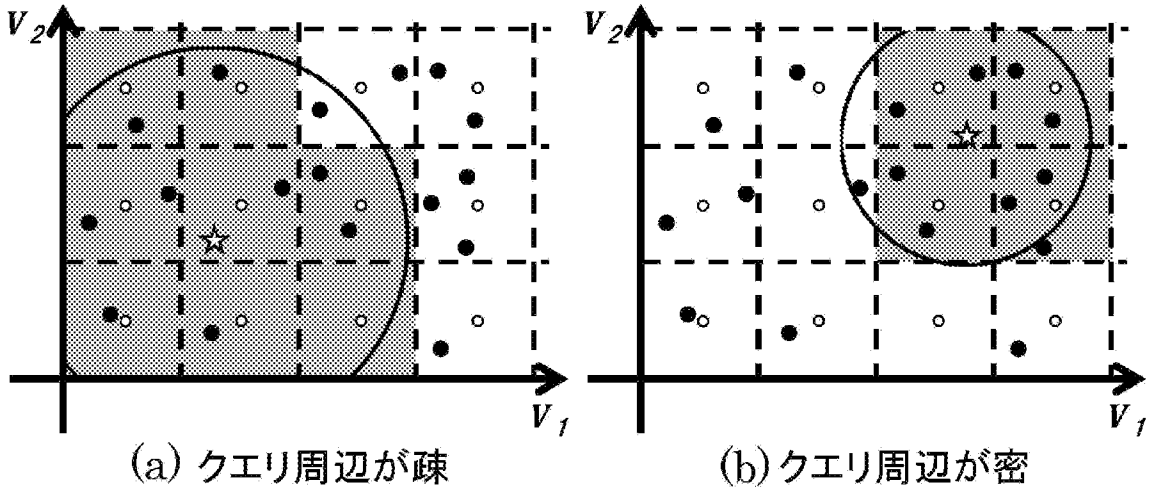
[図6]


60
50
40
30
20
10
0
時間[ms]
0
10
20
30
40
50
60
70
精度[%]
ANN
SH
佐藤らの手法
単一ハッシュ
分割ハッシュ

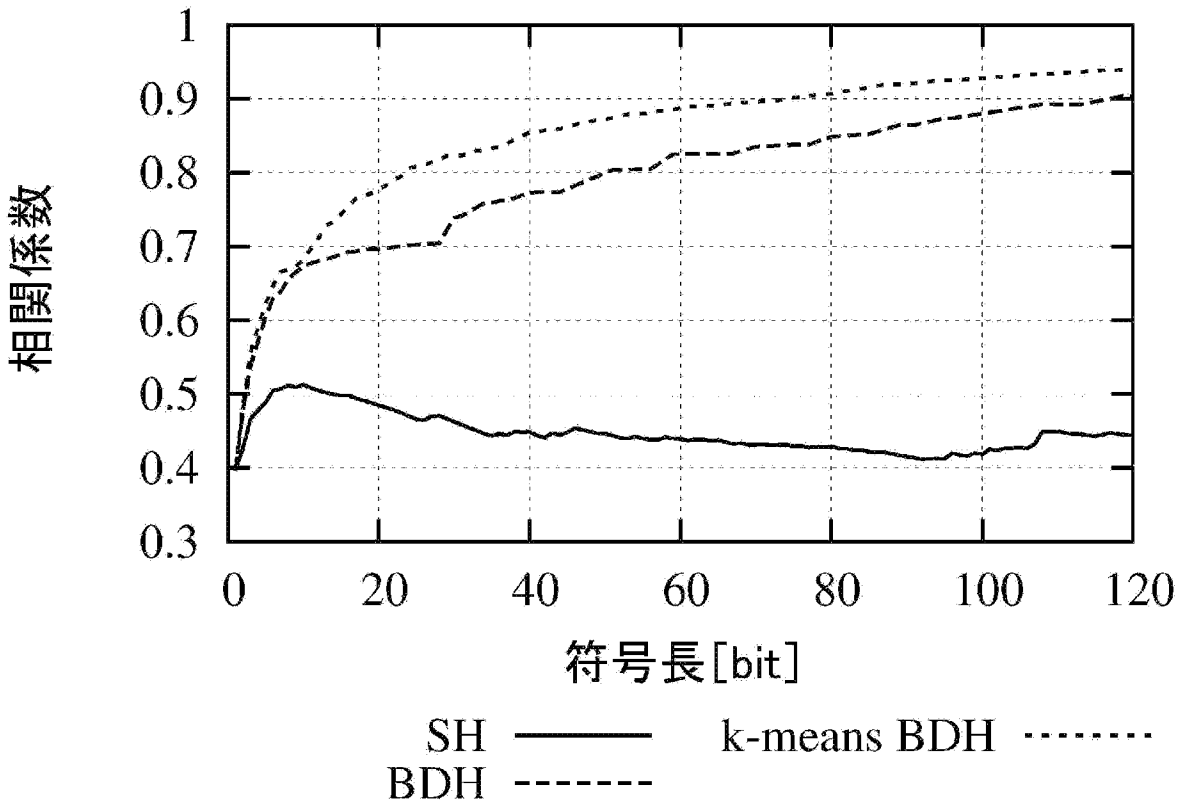
[図7]


確率密度
境界
代表値
等分割
x
0
$\frac{1}{6}\max(x)$
$\frac{1}{3}\max(x)$
$\frac{1}{2}\max(x)$
$\frac{2}{3}\max(x)$
$\frac{5}{6}\max(x)$
適応分割
x
0
C_1
$\frac{C_1+C_2}{2}$
C_2
$\frac{C_2+C_3}{2}$
C_3

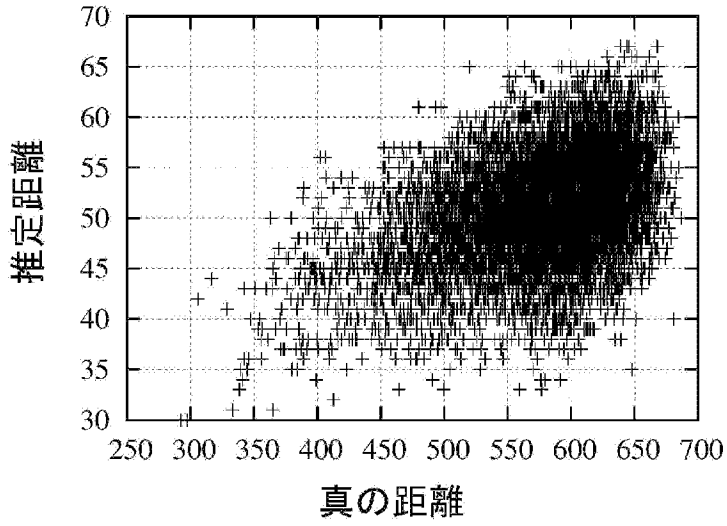
[図8]


V2
V1
(a) クエリ周辺が疎
V2
V1
(b) クエリ周辺が密

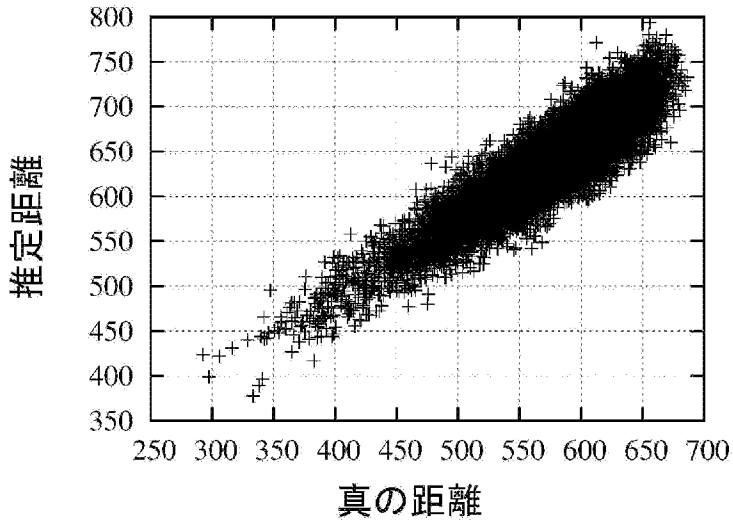
[図9]


1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
相関係数
0
20
40
60
80
100
120
符号長[bit]
SH
k-means BDH
BDH

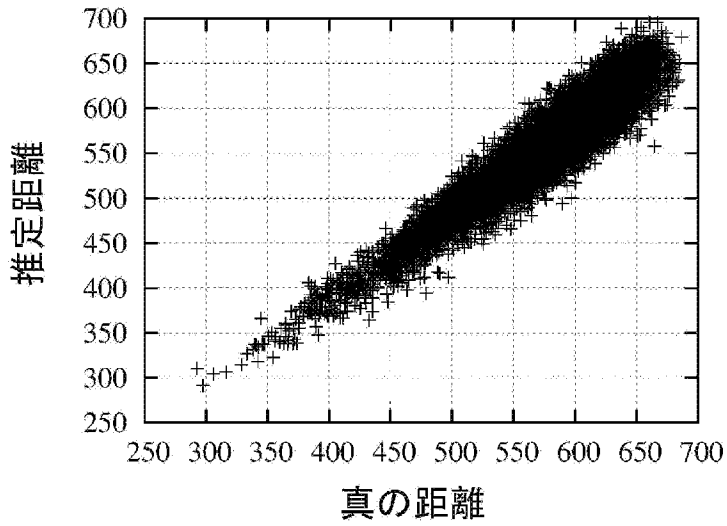
[図10]



(a) SH

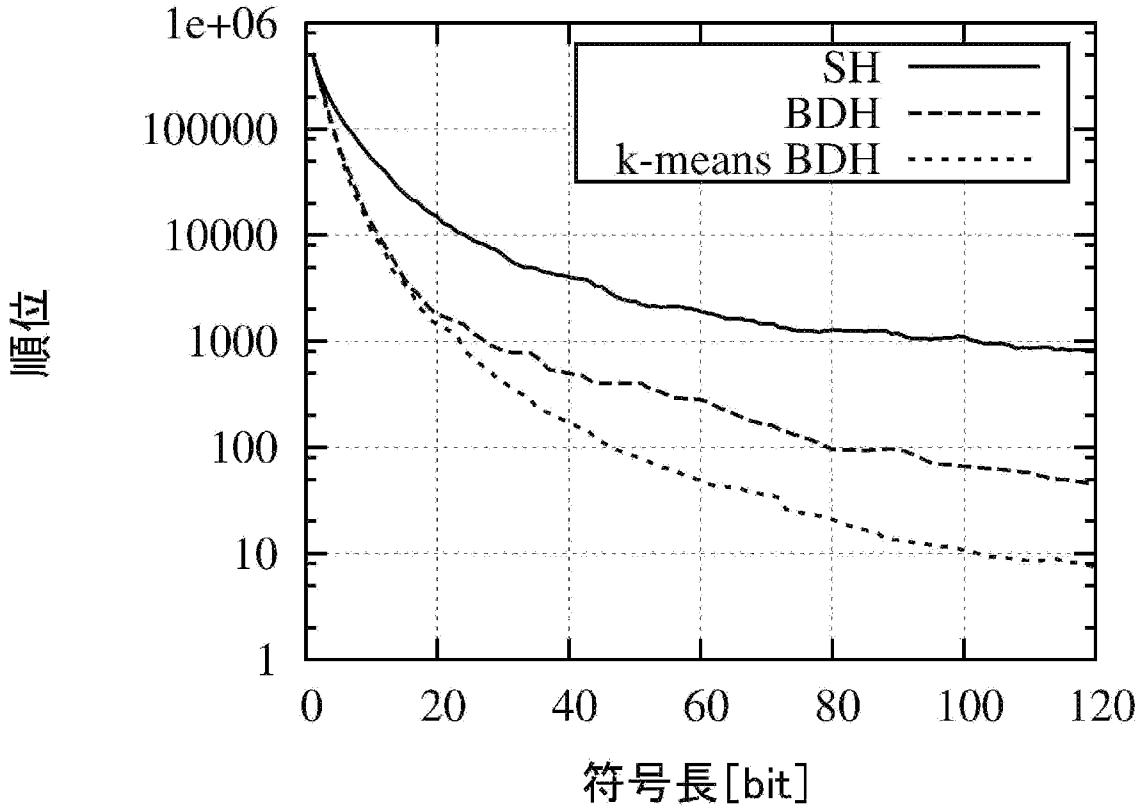


(b) BDH

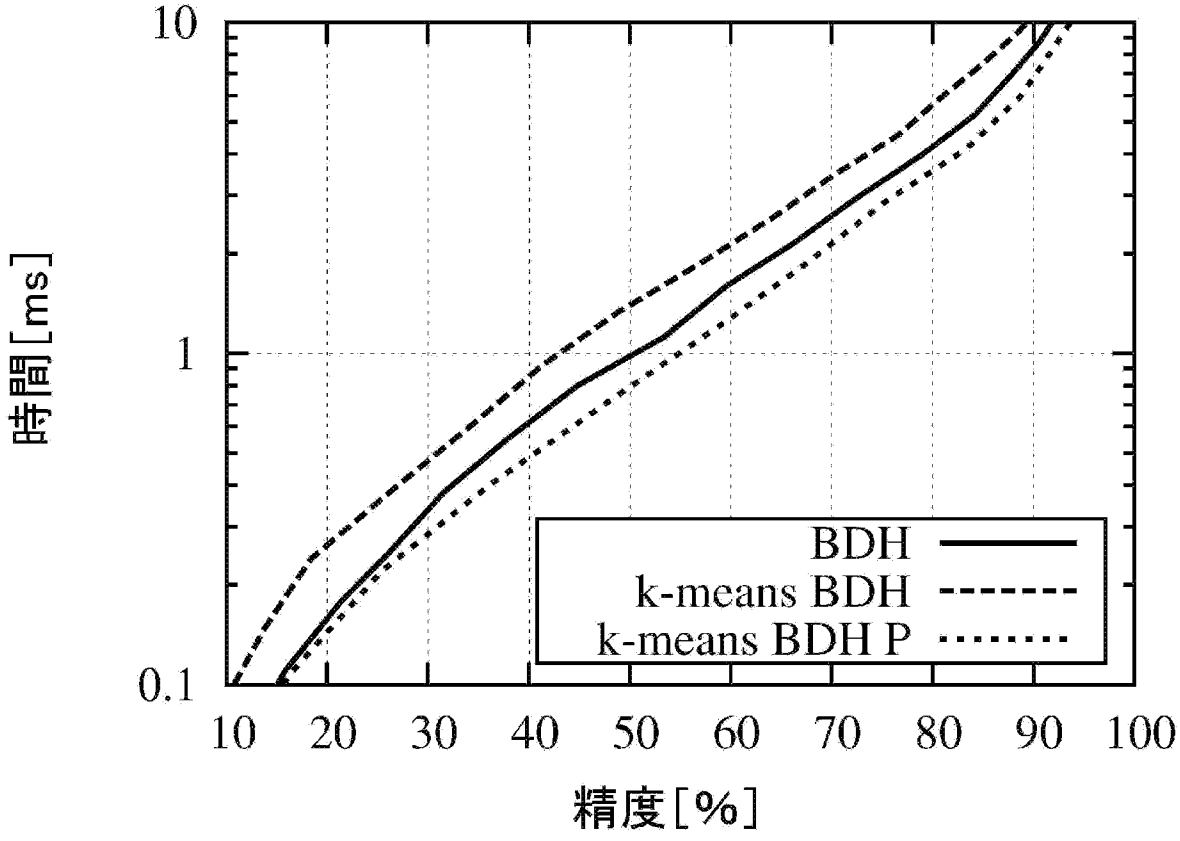


(c) k-means BDH

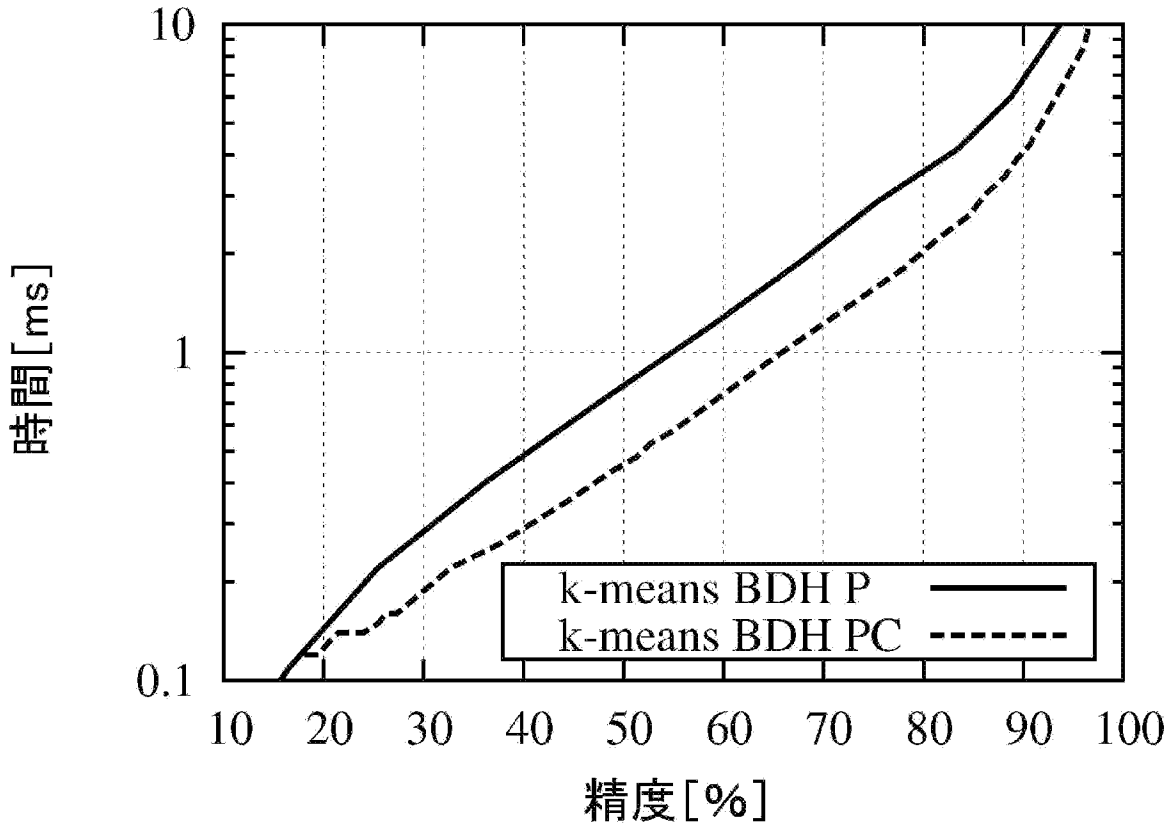
[図11]


1e+06
100000
10000
1000
100
10
1
順位
0
20
40
60
80
100
120
符号長[bit]
SH
BDH
k-means BDH

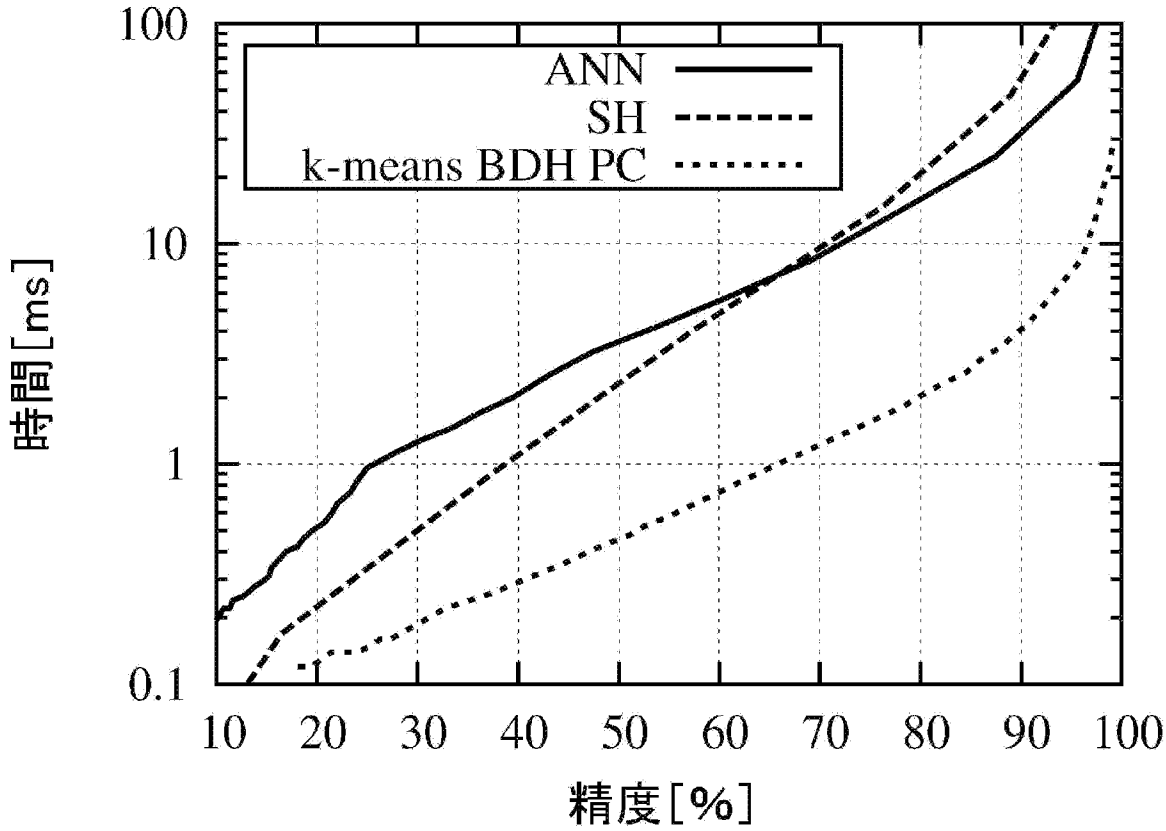
[図12]


10
1
0.1
時間[ms]
10
20
30
40
50
60
70
80
90
100
精度[%]
BDH
k-means BDH
k-means BDH P

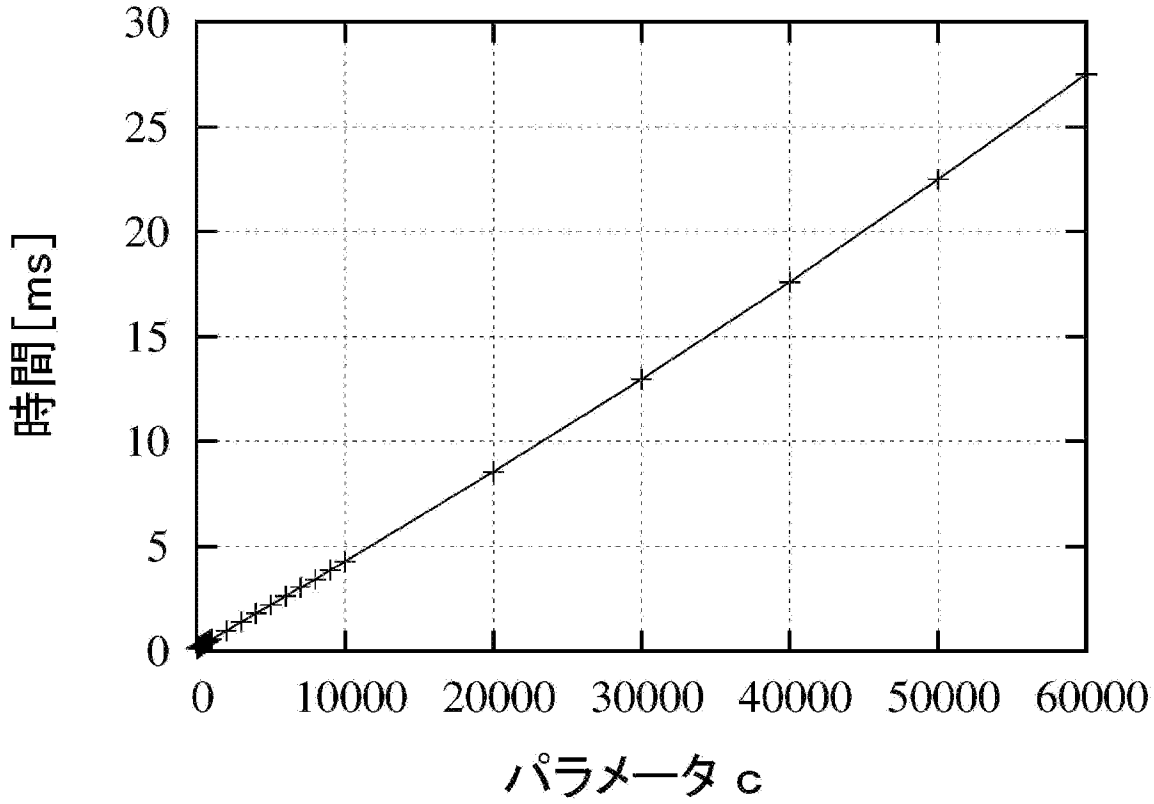
[図13]


10
1
0.1
時間[ms]
k-means BDH P
k-means BDH PC
10 20 30 40 50 60 70 80 90 100
精度[%]

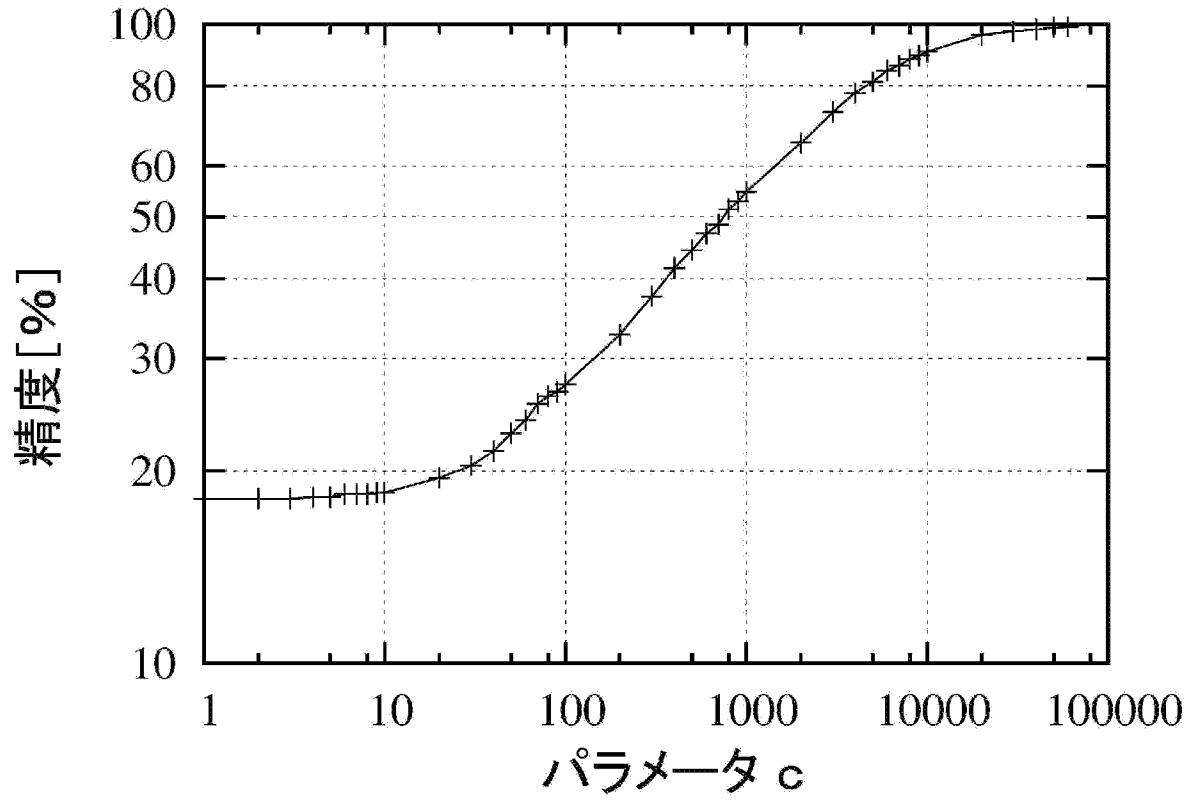
[図14]


100
10
1
0.1
時間[ms]
ANN
SH
k-means BDH PC
10 20 30 40 50 60 70 80 90 100
精度[%]

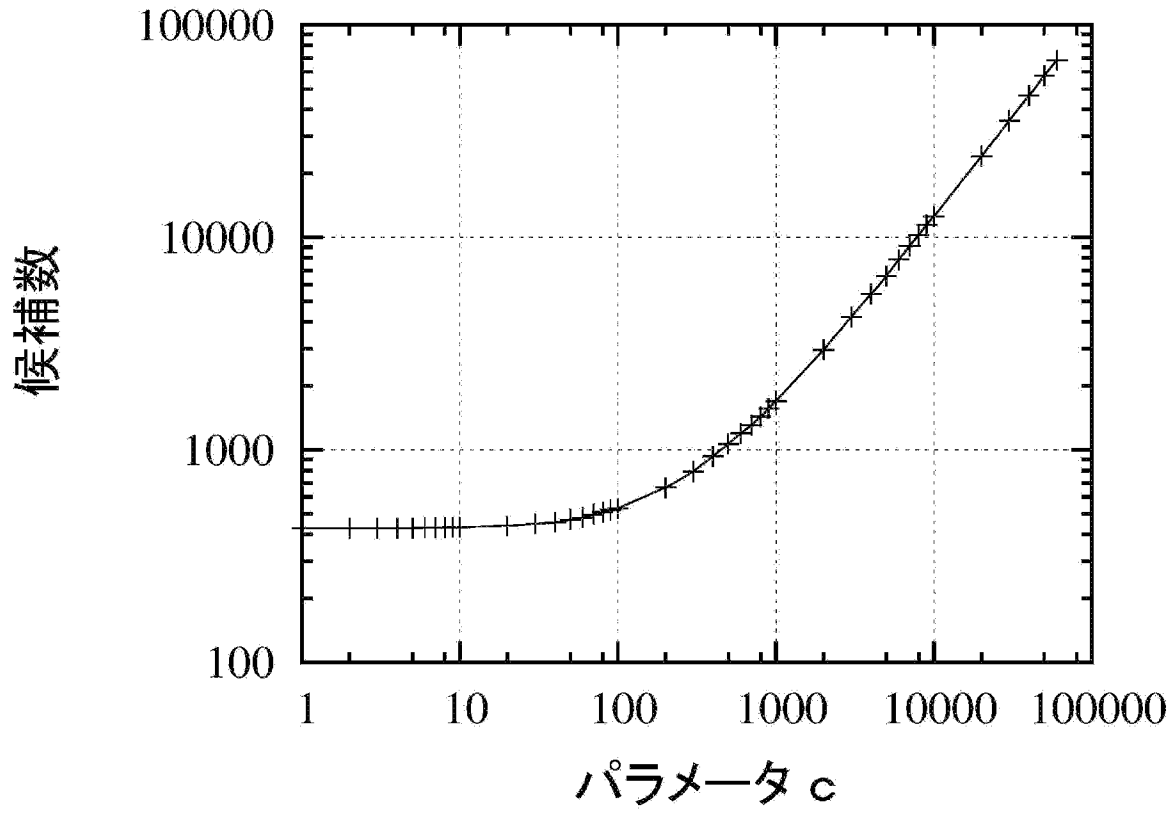
[図15]


30
25
20
15
10
5
0
時間[ms]
0
10000
20000
30000
40000
50000
60000
パラメータ c

[図16]


100
80
60
50
40
30
20
10
精度[%]
1
10
100
1000
10000
100000
パラメータ c

[図17]



[図18]

アルゴリズム１ Search

```
1: for i = 1 to v do
2:    mD_i ← Σ_{b=i+1}^{v} min_j BD_bj
3: end for
4: if Σ_b min_j BD_bj < R then
5:    CircuitBucketsFunction(1, 0)
6: end if
```

アルゴリズム２ CircuitBucketsFunction(i, D)

```
1: if i < v then
2:    for ∀ j do
3:       if D + BD_ij + mD_i < R then
4:          h_i ← j
5:          CircuitBucketsFunction( i + 1, D + BD_ij)
6:       end if
7:    end for
8: else
9:    for ∀ j do
10:      if D + BD_vj < R then
11:         Access a bucket whose hash value is H = {h_1, h_2, ··· , h_v}.
12:      end if
13:   end for
14: end if
```

[図19]

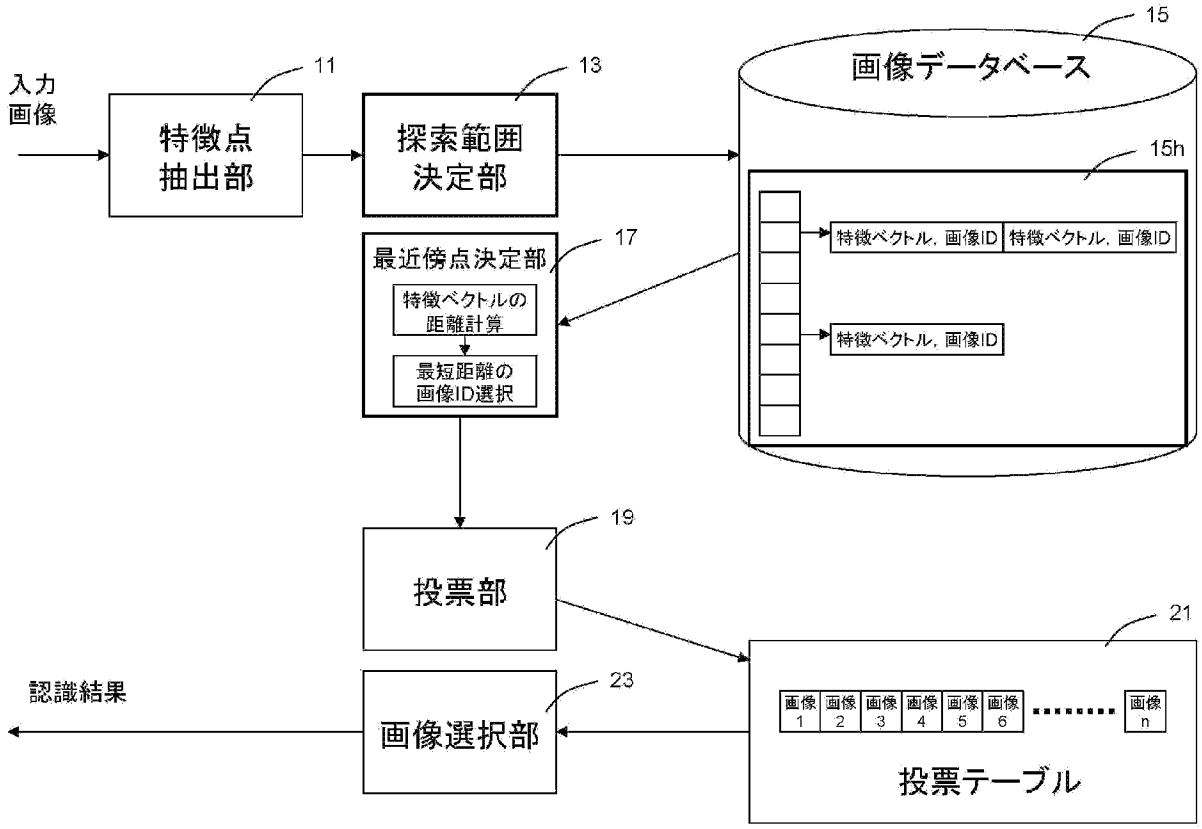
アルゴリズム3 Search_c

for $i = 1$ to v do
  $\mathrm{mD}_i \leftarrow \sum_{b=i+1}^{v} \min_j \mathrm{BD}_{bj}$
  $\mathrm{MD}_i \leftarrow \sum_{b=i+1}^{v} \max_j \mathrm{BD}_{bj}$
end for
$L \leftarrow 0$
$U \leftarrow \sum_b \min_j \mathrm{BD}_{bj} + \Delta$
while The number of candidates is less than c do
  CircuitBucketsFunction_c $(1, 0)$
  $L \leftarrow U$
  $U \leftarrow U + \Delta$
end while

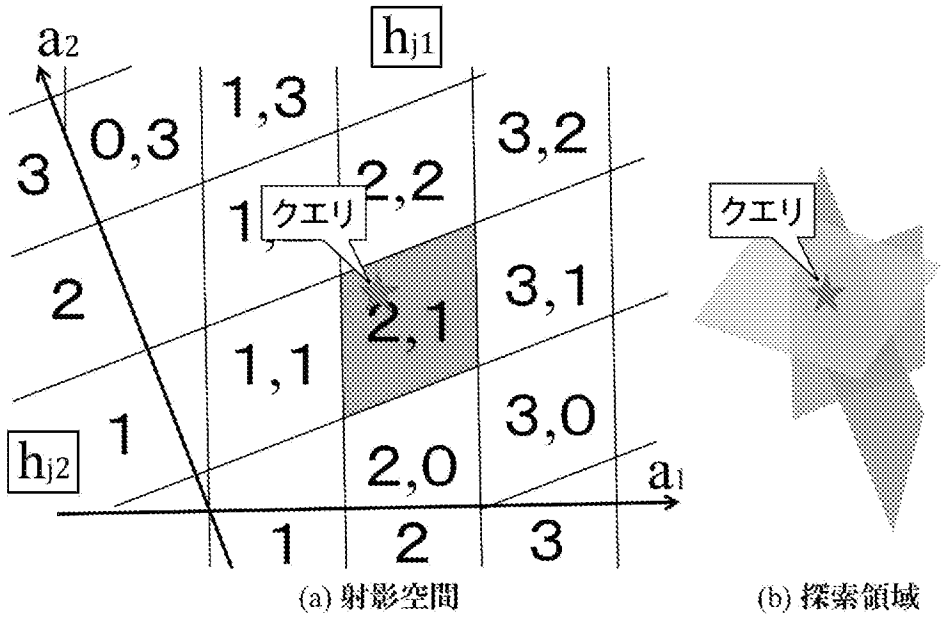
アルゴリズム4 CircuitBucketsFunction_c(i, D)

if $i < v$ then
  for $\forall$ j do
    if $L \leq D + \mathrm{BD}_{ij} + \mathrm{MD}_i$ & $D + \mathrm{BD}_{ij} + \mathrm{mD}_i < U$ then
      $h_i \leftarrow j$
      CircuitBucketsFunction_c$(i + 1, D + \mathrm{BD}_{ij})$
    end if
  end for
else
  for $\forall$ j do
    if $L \leq D + \mathrm{BD}_{ij}$ & $D + \mathrm{BD}_{ij} < U$ then
      Access a bucket whose hash value is $\boldsymbol{H} = \{h_1, h_2, \cdots, h_v\}$.
    end if
  end for
end if

[図20]

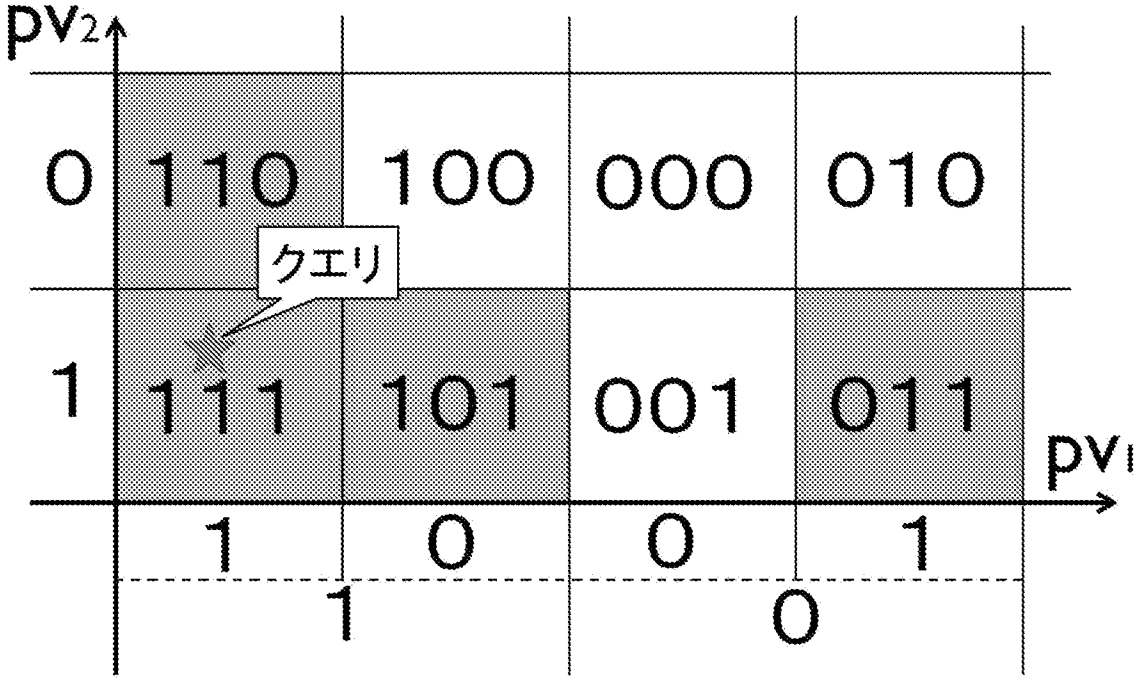


[図21]

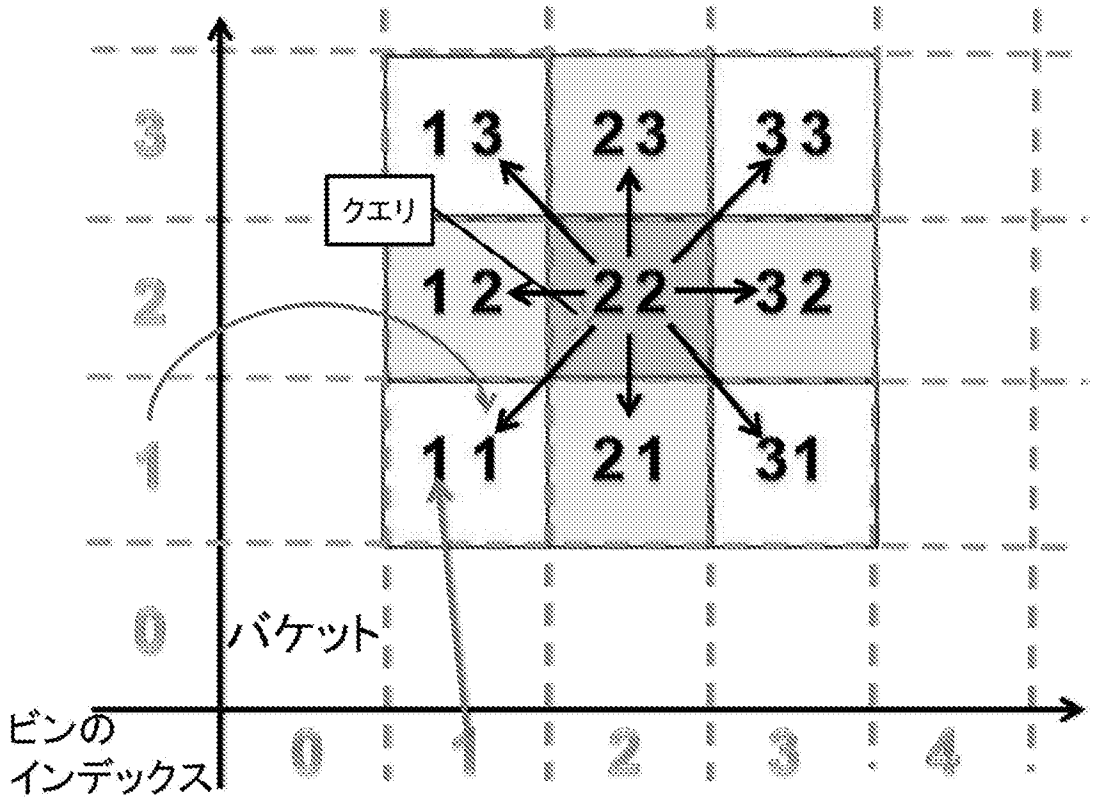


(a) 射影空間 (b) 探索領域

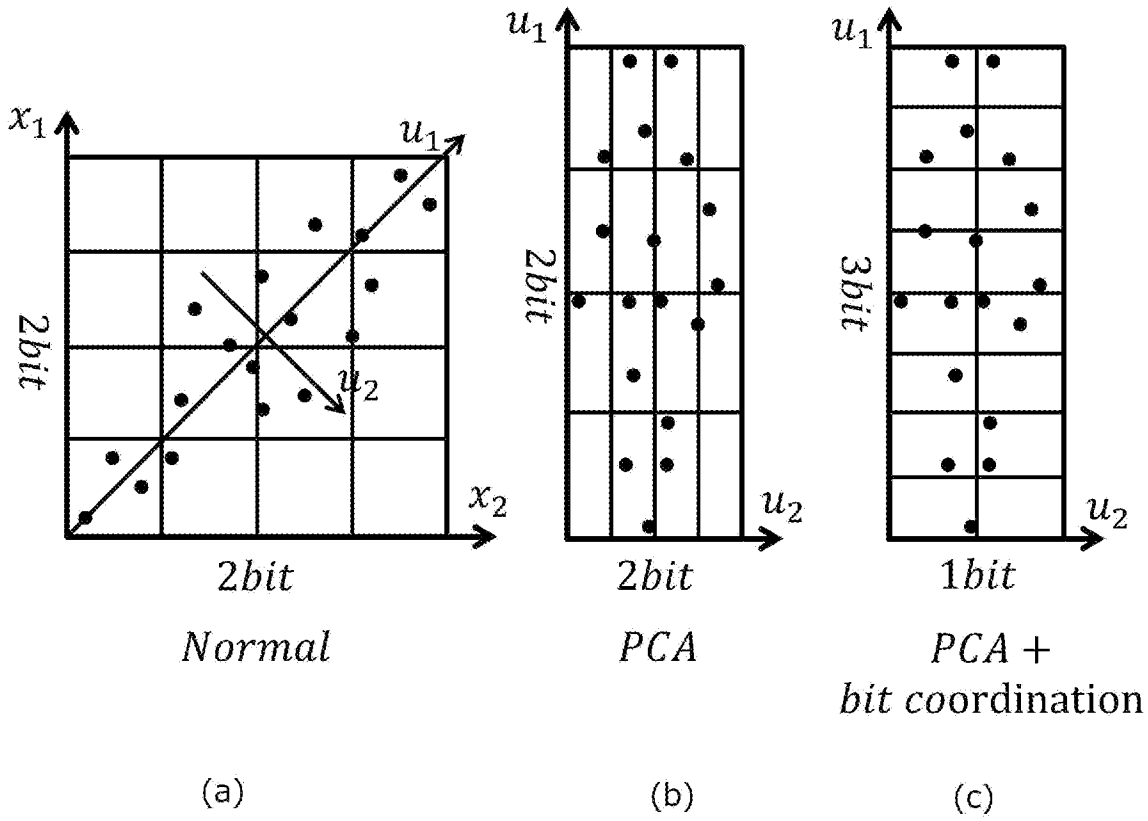
[図22]


pv2
0
1
110
100
000
010
クエリ
111
101
001
011
pv1
1
0
0
1
1
0

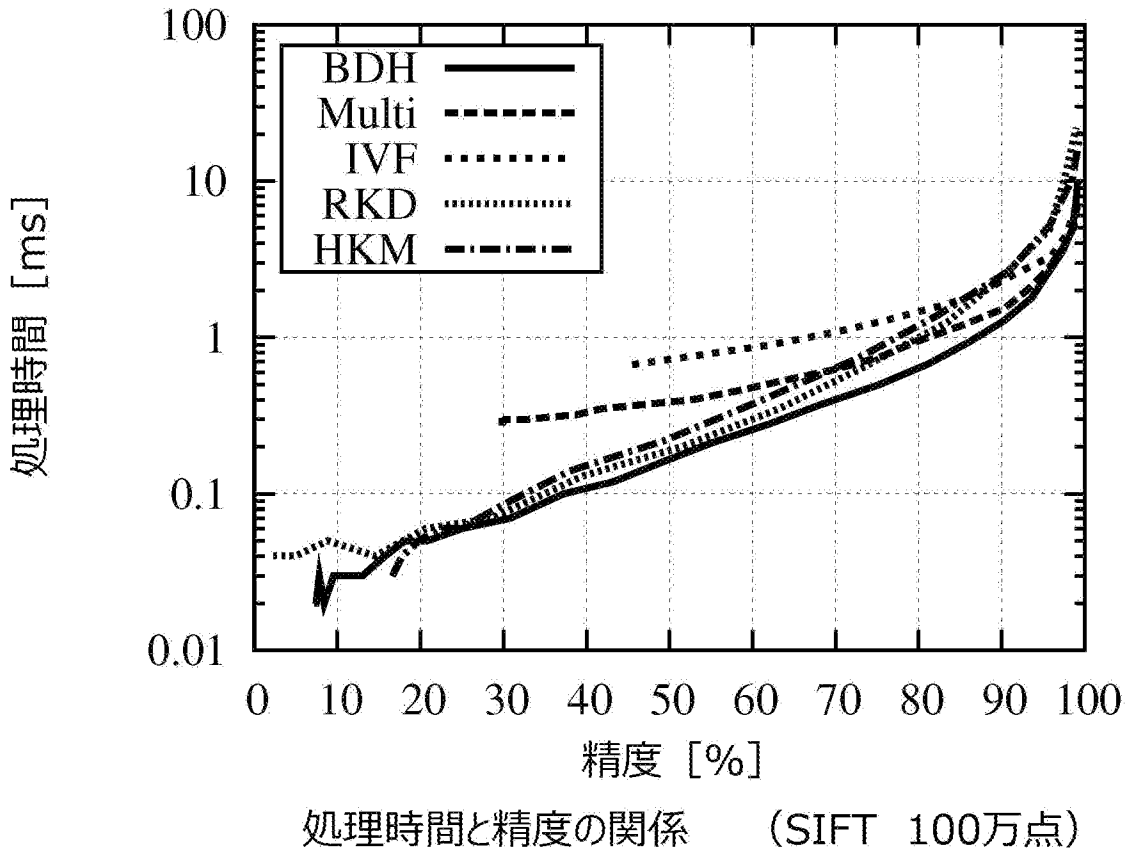
[図23]


3
2
1
0
13
23
33
クエリ
12
22
32
11
21
31
バケット
ビンの
インデックス
0
1
2
3
4

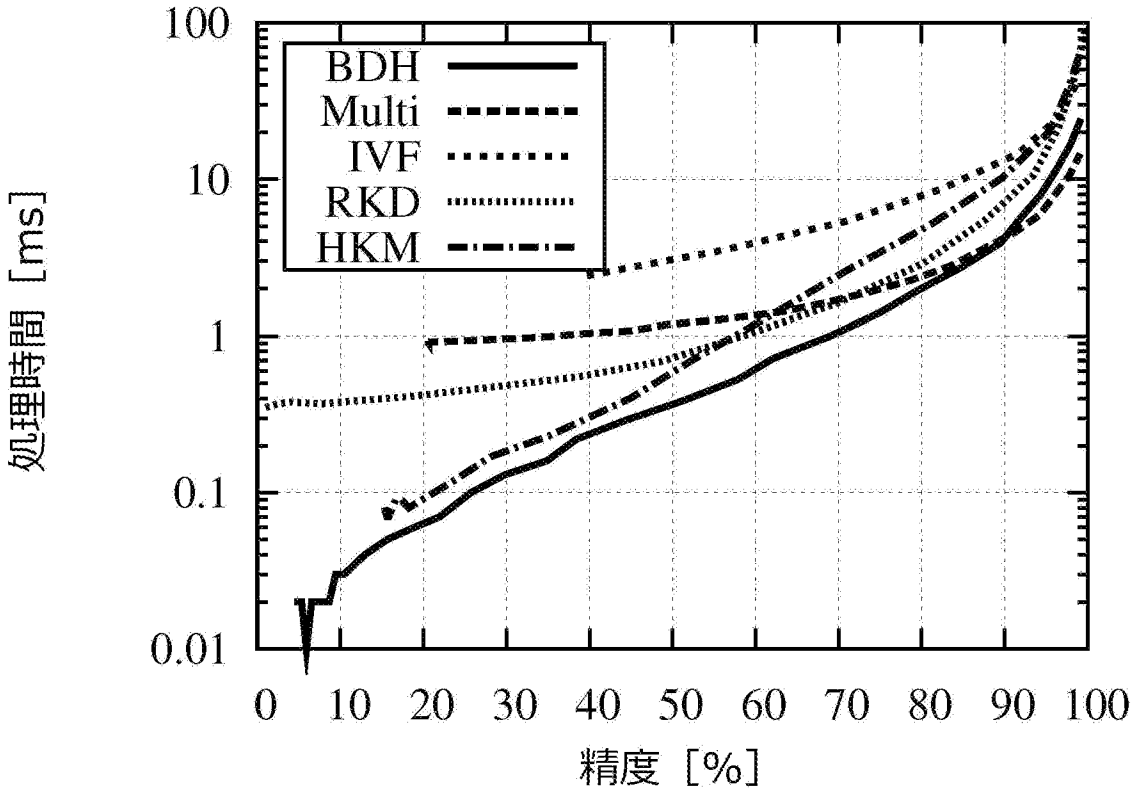
[図24]


x_1
u_1
$2bit$
u_2
x_2
$2bit$
Normal
(a)
u_1
$2bit$
u_2
$2bit$
PCA
(b)
u_1
$3bit$
u_2
$1bit$
PCA +
bit coordination
(c)

[図25]

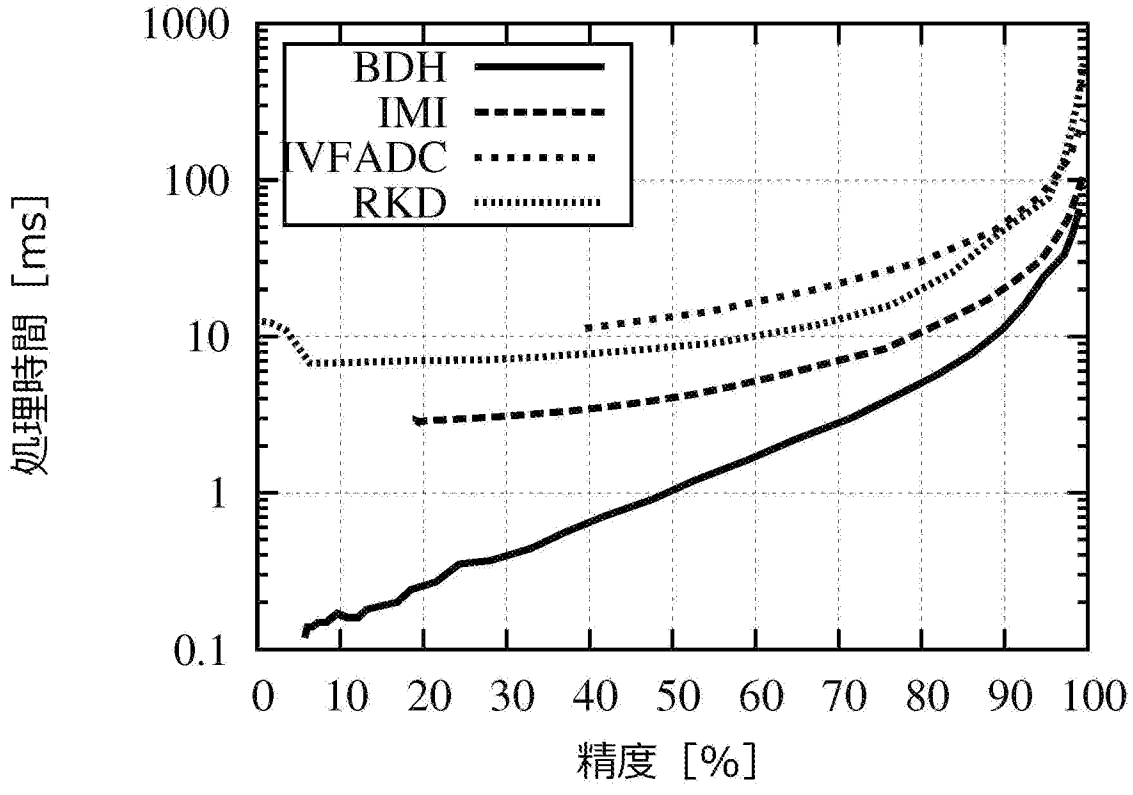

BDH
Multi
IVF
RKD
HKM
処理時間［ms］
100
10
1
0.1
0.01
0
10
20
30
40
50
60
70
80
90
100
精度［%］
処理時間と精度の関係　（SIFT　100万点）

[図26]



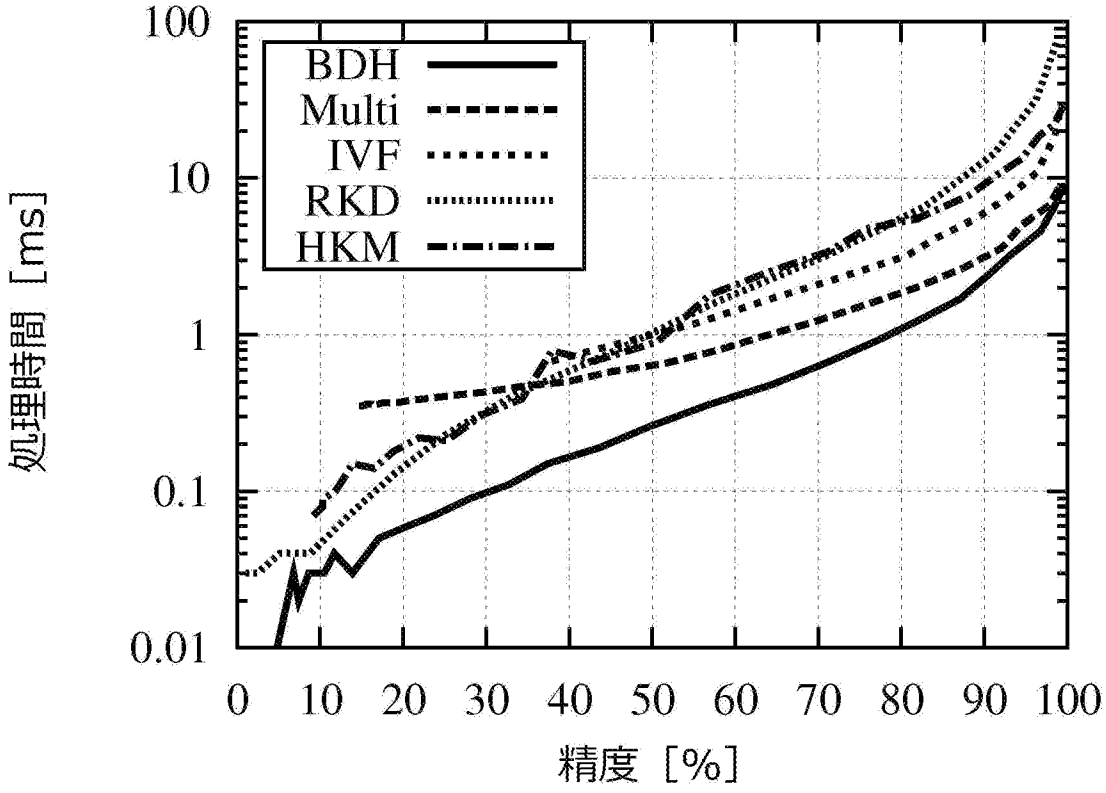
処理時間と精度の関係 （SIFT 1000万点）

[図27]



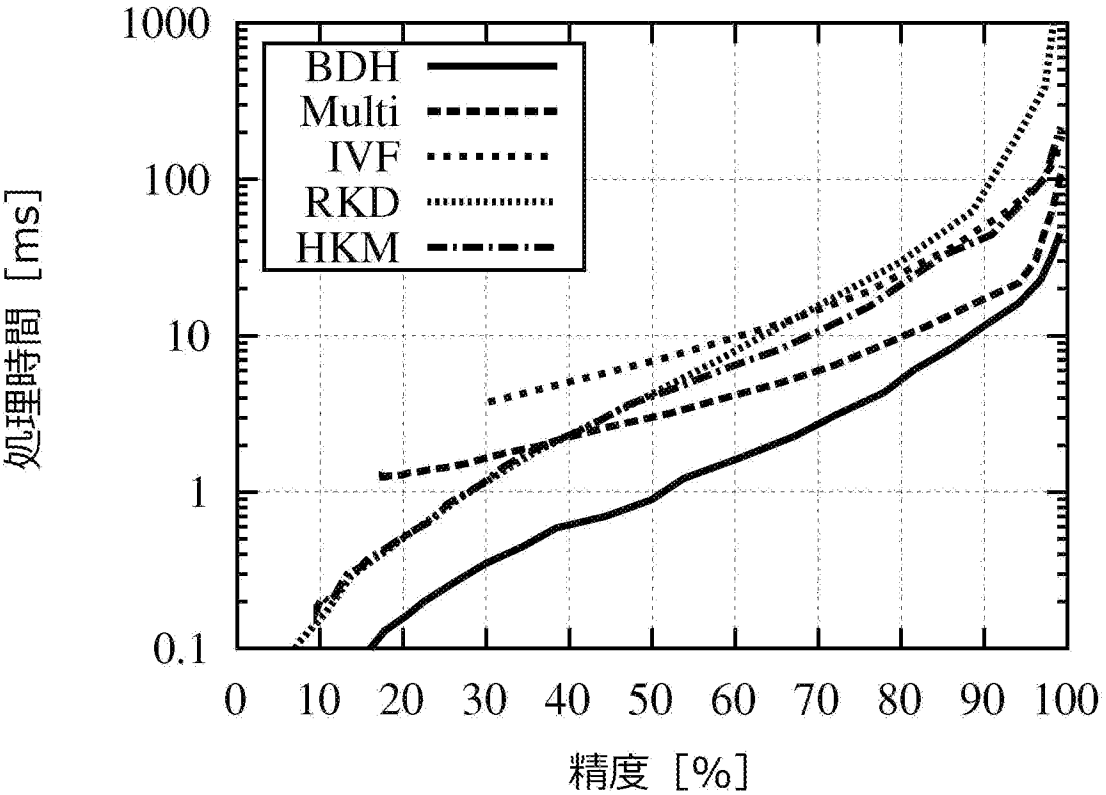
処理時間と精度の関係 （SIFT 1億点）

[図28]



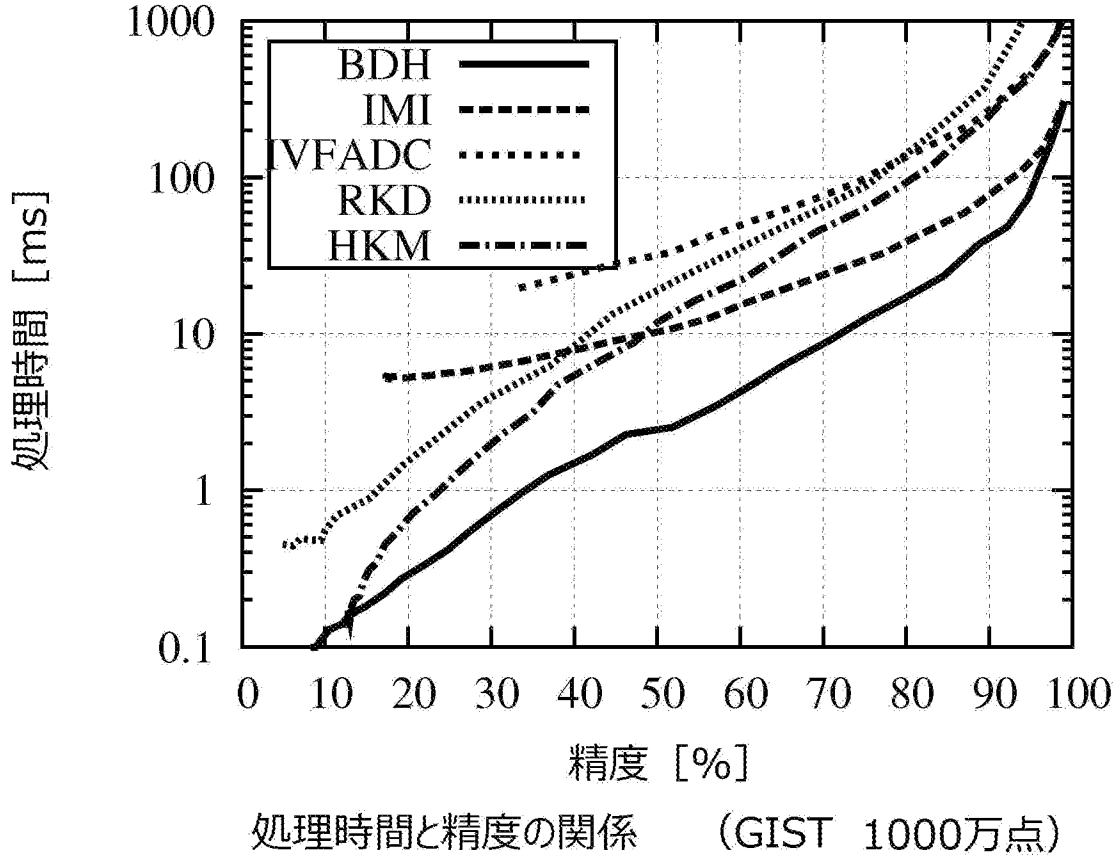
処理時間と精度の関係　（GIST　10万点）

[図29]



処理時間と精度の関係　（GIST　100万点）

[図30]



処理時間と精度の関係　（GIST 1000万点）

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2013/055440

A. CLASSIFICATION OF SUBJECT MATTER
G06F17/30(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F17/30

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Jitsuyo Shinan Koho	1922-1996	Jitsuyo Shinan Toroku Koho	1996-2013
Kokai Jitsuyo Shinan Koho	1971-2013	Toroku Jitsuyo Shinan Koho	1994-2013

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y A	Tomokazu SATO et al., "Gaisan Kyori no Seido Kojo ni yoru Kinji Saikinbo Tansaku no Kosokuka", IPSJ SIG Notes 2011 October [CD-ROM], 15 October 2011 (15.10.2011), pages 1 to 6	1-3,7-8, 11-12 4-6,9-10
Y A	Naoto MASUYAMA et al., "Acceleration of the k-Nearest Neighbor Algorithm by Addition of Termination Conditions in Pattern Recognition Problems", The Transactions of the Institute of Electronics, Information and Communication Engineers, 01 March 2001 (01.03.2001), vol. J84-D-II, no.3, pages 439 to 447	1-3,7-8, 11-12 4-6,9-10

☐ Further documents are listed in the continuation of Box C.

☐ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
26 March, 2013 (26.03.13)	02 April, 2013 (02.04.13)
Name and mailing address of the ISA/ Japanese Patent Office Facsimile No.	Authorized officer Telephone No.

国際調査報告

国際出願番号　ＰＣＴ／ＪＰ２０１３／０５５４４０

Ａ．　発明の属する分野の分類（国際特許分類（ＩＰＣ））

Int.Cl.　G06F17/30(2006.01)i

Ｂ．　調査を行った分野

調査を行った最小限資料（国際特許分類（ＩＰＣ））

Int.Cl.　G06F17/30

最小限資料以外の資料で調査を行った分野に含まれるもの

日本国実用新案公報	１９２２－１９９６年
日本国公開実用新案公報	１９７１－２０１３年
日本国実用新案登録公報	１９９６－２０１３年
日本国登録実用新案公報	１９９４－２０１３年

国際調査で使用した電子データベース（データベースの名称、調査に使用した用語）

Ｃ．　関連すると認められる文献

引用文献のカテゴリー＊	引用文献名　及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求項の番号
Y A	佐藤　智一　他，概算距離の精度向上による近似最近傍探索の高速化，情報処理学会研究報告　２０１１Ｏｃｔｏｂｅｒ［ＣＤ－ＲＯＭ］，2011.10.15，pp.1-6	1-3,7-8,11-12 4-6,9-10
Y A	益山　直人　他，パターン認識問題における終端条件の付加によるｋ近隣法の高速化，電子情報通信学会論文誌，2001.03.01，第J84-D-II巻　第３号，pp.439-447	1-3,7-8,11-12 4-6,9-10

☐　Ｃ欄の続きにも文献が列挙されている。

☐　パテントファミリーに関する別紙を参照。

＊　引用文献のカテゴリー

「Ａ」特に関連のある文献ではなく、一般的技術水準を示すもの

「Ｅ」国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの

「Ｌ」優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献（理由を付す）

「Ｏ」口頭による開示、使用、展示等に言及する文献

「Ｐ」国際出願日前で、かつ優先権の主張の基礎となる出願の日の後に公表された文献

「Ｔ」国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの

「Ｘ」特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの

「Ｙ」特に関連のある文献であって、当該文献と他の１以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの

「＆」同一パテントファミリー文献

国際調査を完了した日	国際調査報告の発送日
２６．０３．２０１３	０２．０４．２０１３

国際調査機関の名称及びあて先
日本国特許庁（ＩＳＡ／ＪＰ）
郵便番号１００－８９１５
東京都千代田区霞が関三丁目４番３号

特許庁審査官（権限のある職員）　５Ｍ　４６７８
久々宇　篤志
電話番号　０３－３５８１－１１０１　内線　３５９９