5H-06

GPU を用いた類似画像検索の高速化

草村 優太 † 小澤 佑介 †† 天笠 俊之 ††† 北川 博之 †††

† 筑波大学情報学群情報科学類

†† 筑波大学大学院システム情報工学研究科 ††† 筑波大学システム情報系情報工学域

1 はじめに

画像集合の中からクエリ画像と視覚的に類似した画像を見つけ出す処理を，類似画像検索と呼ぶ．近年の類似画像検索は画像から抽出される特徴量を用いる手法が主流である．その際，一つの画像は数百から数千のベクトルで表現される．このため，類似検索処理における特徴量の比較演算には多大な処理コストを要するという問題がある．特徴量を圧縮することで処理効率を向上させるアプローチが検討されているが，大量の画像を効率的に扱うには更なる高速化が必要である．そこで本研究では，GPU を利用した類似画像検索の高速化手法を提案する．具体的には，特徴量の圧縮に LSH を用い，各特徴量の近傍探索を並列に実行することで高速化を達成する．

2 基本事項

2.1 局所特徴量と類似画像検索

局所特徴量とは，近年画像検索で標準的に用いられる特徴量の総称である．局所特徴量には SIFT 特徴量や SURF 特徴量など，いくつかの種類が存在するが，本稿では画像認識の分野で最も基本的な特徴量である SIFT 特徴量を用いる．

SIFT 特徴量とは，一つの画像を複数の 128 次元ベクトルで表現したものである．その抽出の際には，まず特徴点の検出を行う．特徴点は画像中の濃淡の差等を考慮して検出される．特徴点の数は画像によって異なるが，一般に一つの画像から数百から数千の特徴点が得られる．次に，各特徴点における特徴ベクトルを抽出する．特徴ベクトルは特徴点周辺の濃淡の勾配に基づいて計算され，一つの特徴点に対して 128 次元のベクトルで表現される．結果として一つの画像は数百から数千本程度の 128 次元ベクトルで表現される．

GPU Acceleration of Content-based Image Retrieval
Yuta Kusamura†, Yusuke Kozawa††,
Toshiyuki Amagasa††† and Hiroyuki Kitagawa†††
†College of Information Science, University of Tsukuba
††Gradurate School of Systems and Information Engineering, University of Tsukuba
†††Faculty of Engineering, Information and Systems, University of Tsukuba

SIFT 特徴量を用いた，単純な類似画像検索は以下の様に実現することができる．その処理は，データベース構築とクエリ処理に分けられる．データベース構築では，まず，画像集合中の各画像から SIFT 特徴量を抽出する．次に，抽出された 128 次元の特徴ベクトルと，そのベクトルが抽出された画像 ID のタプルを，データベース $\boldsymbol{D}$ に格納する．クエリ画像が入力された際，クエリ画像から SIFT 特徴量を抽出する．ここで，クエリ中の k 番目の特徴ベクトルを $\boldsymbol{q}_k$，$\boldsymbol{D}$ 中の l 番目のタプルを $(\boldsymbol{f}_l, id_l)$ とすると，$\boldsymbol{q}_k$ に対して最も近い $\boldsymbol{D}$ 中のベクトルを検索しそれに対応付けられた画像 ID を取得する．すなわち，

$$\{id_i \mid \forall j : (\boldsymbol{f}_j, id_j) \in \boldsymbol{D} \wedge dist(\boldsymbol{q}_k, \boldsymbol{f}_i) \leq dist(\boldsymbol{q}_k, \boldsymbol{f}_j)\} \quad (1)$$

ただし，$dist(\boldsymbol{a}, \boldsymbol{b})$ は，ベクトル $\boldsymbol{a}$，$\boldsymbol{b}$ 間の距離である．クエリ内の全ての特徴ベクトルに対して式 (1) の検索を行い，最も出現頻度が多い画像 ID が検索結果となる．

2.2 GPGPU

GPU (Graphics Processing Unit) は従来グラフィック処理に用いられ，高い並列性を持つ点が特徴である．その GPU の高い並列性に着目して，高速な並列処理を行うことを GPGPU (General-purpose Computing on Graphics Processing Units) と呼ぶ．GPGPU では，汎用的な計算が可能であり，様々な分野で研究がなされている．本稿では，NVIDIA 社の GPU 及び GPU 開発環境である CUDA を用いる．

GPU での処理は，スレッド，ブロック，グリッドの 3 つに分けられる．スレッドが GPU での処理の最小単位であり，複数のスレッドをまとめたものをブロックと呼ぶ．さらにブロックをまとめたものをグリッドと呼ぶ．

GPU は CPU と構成が大きく異なる．GPU には，レジスタ，シェアードメモリ，グローバルメモリ等，性質の異なる複数のメモリが搭載されており，アクセス速度やサイズが異なる．また，GPU はスカラプロセッサと呼ばれる演算装置が数百個から数千個搭載されている．複数のスカラプロセッサとシェアードメモリをまとめたものをストリーミングマルチプロセッサと呼び，これが実際の計算処理単位となる．

このように，GPU には CPU とは異なる性質が多くある．そのため，GPU の並列処理性を活かすには，GPU の構造を理解した上での適切なプログラムの設計が必要となる．

3 関連研究

SIFT 特徴量を用いた類似画像検索の効率化にはいくつかの手法がある．Zhu らの手法 [2] では，SIFT 特徴量をベクトル量子化し，それを用いて近似最近傍探索を行うことにより効率化を図っている．また，GPU を用いて検索を効率化している研究もなされている．Cevahir らの手法 [1] では，SIFT 特徴量を階層的にクラスタリングし，クラスタ内で近似最近傍探索を行う．このとき，木の探索の処理と，クラスタ内での近傍探索を GPU を用いて並列に行うことにより，類似画像検索の効率化を図っている．Chandrasekhar らのサーベイ [3] では，複数の SIFT 特徴量の圧縮方法の比較を行っている．このように，SIFT 特徴量を圧縮して検索に用いることも検索の効率化に対して有効である．

4 提案手法

本稿では，類似画像検索の処理のうち，入力としてクエリが与えられてから，結果を出力するまでの処理を高速化する．すなわち，検索処理時点では，画像集合に基づいたデータベースはすでに構築済であるとする．この検索処理の効率化のために，LSH による特徴量の圧縮と GPU による並列化を行う．

4.1 LSH (Locality Sensitive Hashing)

特徴量の圧縮のために LSH (Locality Sensitive Hashing) を用いる．LSH とは，類似したベクトルを高い確率で類似したハッシュ値に変換するハッシュ法の一つである．LSH はそのハッシングの方法によりいくつかの種類に分類される．本稿では，コサイン類似度に対する LSH の手法に基いて特徴量の変換を行う．

コサイン類似度に対する LSH では，ベクトル空間における k 個のランダムな超平面を作成し，ベクトルがそれぞれの超平面に対してどちら側にあるかによって各ビットの値を決定している．すなわち，圧縮するベクトルを $\boldsymbol{v} = (v_1, v_2, ..., v_d)$ とし，k 個のランダムな超平面 (法線ベクトル) を $\boldsymbol{u} = (u_1, u_2, ..., u_d)$ としたとき，i 番目のビットを決定するハッシュ関数 h_i は，

$$h_i(\boldsymbol{v}) = \begin{cases} 1 & (\boldsymbol{v} \cdot \boldsymbol{u} \geq 0) \\ 0 & (\boldsymbol{v} \cdot \boldsymbol{u} < 0) \end{cases} \quad (2)$$

で表される．

4.2 データベースの構築

以下の手順によってデータベースを構築すると共に，GPU 上のデバイスメモリにロードしておく．はじめに，2.1 節で説明した手順に従い，画像集合中の各画像に対して特徴ベクトルと画像 ID のタプル $(\boldsymbol{f}_i, id_i)$ を抽出する．次に，各タプルの特徴ベクトルに LSH を適用し，$\boldsymbol{f}'_i = h(\boldsymbol{f}_i)$ を計算した上で，ビット列と画像 ID のタプル $(\boldsymbol{f}'_i, id_i)$ をデータベース $\boldsymbol{D'}$ に格納する．得られたデータベースから，以下の三つの配列を作成し，GPU に転送する．

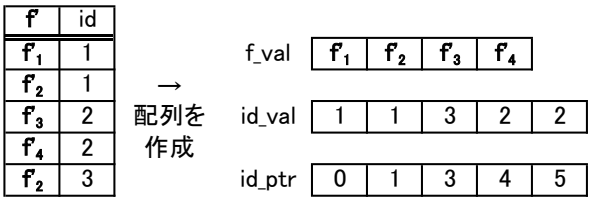

図 1 データベースからの配列の作成.

- f_val: $\{\boldsymbol{f}'_i \mid (\boldsymbol{f}'_i, id_i) \in \boldsymbol{D'}\}$ を値の昇順でソートした配列
- id_val: $\boldsymbol{D'}$ 中の全ての id_i を f_val の順番で格納した配列
- id_ptr: $id\_ptr[i]$ が id_val の先頭レコードからのオフセットを示す配列

図 1 は，データベースから三つの配列を作成する例である．

4.3 GPU を用いた検索

入力としてクエリ画像が与えられると，まず特徴ベクトルを抽出する．次に，特徴ベクトルを GPU に転送し，LSH を用いてビット列に変換する．この処理は，各特徴ベクトルに GPU の一つのスレッドを割り当て，並列に計算を行うことで高速化を行う．

LSH を用いているため，クエリの各ビット列と完全に一致するビット列を検索することで，近傍検索が可能である．そこで，f_val から，一致するビット列を二分探索を用いて検索し，id_val と id_ptr から対応する画像 ID を得る．この検索を，各ビット列に一つのスレッドを割り当てることで並列に行い，最も得られた回数が多い画像 ID を検索結果とする．

5 まとめ

本稿では，SIFT 特徴量に基づく類似画像検索の高速な計算手法を提案した．その際，LSH を用いた特徴量の圧縮と，GPU による並列化を行った．今後は，評価実験を通して本手法の有用性を検証する予定である．

謝辞

本研究は，JSPS 科研費 (26280037) の支援によるものである．

参考文献

[1] Ali Cevahir, Junji Torii, GPU-Enabled High Performance Online Visual Search with High Accuracy, ISM 2012
[2] Cai-Zhi Zhu, Xiao Zhou, Shin'ichi Satoh, Bag-of-Words Against Nearest-Neighbor Search for Visual Object Retrieval, ACPR 2013
[3] Vijay Chandrasekhar, Mina Makar, Gabriel Takacs, David Chen, Sam S. Tsai, Ngai-man Cheung, Radek Grzeszczuk, Yuriy Reznik, Bernd Girod, Survey of SIFT Compression Schemes, WMMP 2010