

E-05

GPU を用いた高速な非剛体位置合わせの検討

A Preliminary Implementation of Fast Nonrigid Registration on the GPU

池田 圭[†] 伊野 文彦[†] 萩原 兼一[†]

Kei Ikeda Fumihiko Ino Kenichi Hagihara

1. はじめに

画像位置合わせとは、患者の体位や呼吸により生じる位置ずれを画像間で補正する技術である。特に、軟組織を対象とする技術を非剛体位置合わせと呼ぶ。多くの位置合わせアルゴリズムは画像間の類似度を定義し、類似度をコスト関数とする最適化問題に帰着している。

画像のモダリティに依存しないコスト関数として、情報理論に基づく相互情報量が広く使われている。その計算は輝度値の結合ヒストグラム（以下、ヒストグラム）を必要とする。しかし、ヒストグラムの作成は計算量の多い処理であるため高速化が必要である。

そこで近年、GPU（Graphics Processing Unit）向け統合開発環境 CUDA（Compute Unified Device Architecture）¹⁾ を用いて、ヒストグラム作成を並列処理する研究が行われている。単純な並列化では、複数のスレッドが単一のヒストグラムを共有することになる。この場合、特定のメモリ番地に複数のスレッドが同時に書き込む可能性があるため、アトミック演算を用いて排他的にヒストグラムを更新する必要がある。この書き込みは逐次処理されてしまうため、性能ボトルネックとなる。

そこで、Shams ら²⁾ はスレッドごとにヒストグラムのコピー（局所ヒストグラム）を保持し、アトミック演算を使うことなく輝度値を計数している。その後、局所ヒストグラムを縮約（reduction）することで1つのヒストグラムを得る。しかし、GPU は多数のスレッドを実行して実効性能を向上する。したがって、局所ヒストグラムの数が多く、それらの初期化および結合時のオーバーヘッドが大きい。

そこで本稿は、非剛体位置合わせを高速化するために、高速なヒストグラム作成手法を提案する。提案手法は、(1) アトミック演算に起因する書き込みの逐次化、および (2) 局所ヒストグラムに起因するオーバーヘッドがトレードオフの関係にあることに着目し、スレッドのまとまり（スレッド束）ごとに局所ヒストグラムを保持する。スレッド束のサイズ、すなわちスレッド数を実験的に調整することにより、(1) および (2) の両立を図る。

2. CUDA による結合ヒストグラム作成

ヒストグラムの作成対象となる画像を R および F とする。ま

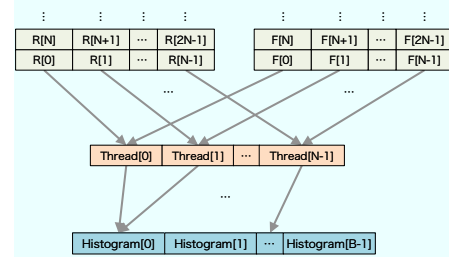


図 1 ヒストグラム作成の単純な並列化

た、画像 R および F の座標 (x, y, z) にあるボクセルの輝度値をそれぞれ $R(x, y, z)$ および $F(x, y, z)$ とする。このとき、ヒストグラムを構成する各ビンは、輝度値の組 $\langle R(x, y, z), F(x, y, z) \rangle$ で表せる。

R および F がそれぞれ n 階調であるとき、ヒストグラムのビン数 B は n^2 である。一般に、医用画像では $n \geq 128$ であり、 B は大きい。したがって、容量の小さい共有メモリにヒストグラムの全体を格納することはできない。

以下、ヒストグラム作成の単純な並列化について考える（図 1）。スレッドの数を N とする。各スレッドは R および F から輝度値を 1 つ読み込み、ヒストグラムを更新する。輝度値の読み込みでは、各スレッドが異なるメモリ番地を参照し、メモリ・コアレスシングが可能である。一方、ヒストグラムの更新では、複数のスレッドが同一のメモリ番地に書き込む可能性があり、アトミック演算を必要とする。

特に、輝度値の分布に偏りがある場合、特定のメモリ番地への書き込みが集中する可能性がある。アトミック演算を回避するためには、書き込み先のメモリ番地をスレッドごとに区別すればよい。Shams²⁾ らの局所ヒストグラムは、そのための解決策の 1 つである。

3. 高速なヒストグラム作成手法

提案手法は、ワーブやスレッドブロックなどのスレッド束ごとに局所ヒストグラムを作成し、それらを縮約することでヒストグラムを作成する（図 2）。以降、スレッド束のサイズを S とする。

各スレッドは、 R および F から輝度値を読み込み、自身が属するスレッド束の局所ヒストグラムを更新する。スレッド束ごとに書き込み領域が用意されているため、スレッド束間で書き込み先が重複することはない。しかし、スレッド束内での重複は排除できないため、更新時にはアトミック演算が必要である。

1 章で挙げた (1) および (2) はサイズ S に依存する。さらに、両者はトレードオフの関係にある。したがって、適切

[†] 大阪大学大学院情報科学研究科コンピュータサイエンス専攻
Department of Computer Science, Graduate School of Information Science and Technology, Osaka University

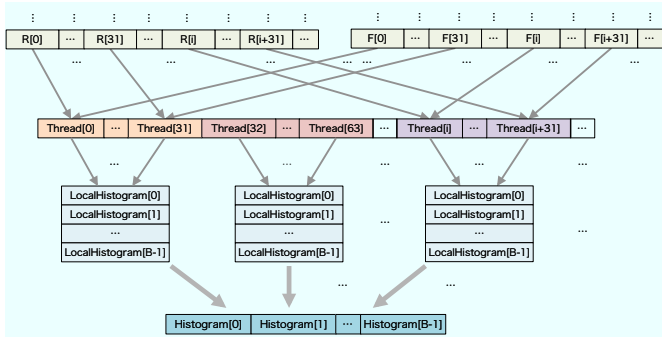


図 2 提案手法によるヒストグラム作成

なサイズ S を決定することにより、両者の削減を両立したヒストグラム作成が可能である。本手法は、トレードオフ・ポイントとなるサイズ S を、ワーブサイズの倍数から決定する。この理由は、メモリアクセスがワーブ（32 スレッド）単位で実行されるからである。

本手法の特長は、適切なサイズ S の決定により、ヒストグラム作成を高速化できることである。ただし、適切なサイズ S は輝度値の分布に依存するため、分布ごとの適切なサイズ S を実験的に決定する必要がある。

4. 実験

スレッド束の適切なサイズ S の決定により、ヒストグラム作成が高速化できることを実験により確認する。入力画像は $512 \times 512 \times 295$ ボクセルの画像 2 組である。図 3(a) にスライスの例を示す。また、図 3(b) に図 3(a) の輝度値の分布を示す。実験ではサイズ S ごとに、ヒストグラム作成時間を計測した。表 1 および図 4 に、実験環境および実験結果を示す。

実験結果から、図 3(b) に示した輝度値の分布では、サイズ $S = 32$ がトレードオフ・ポイントであることがわかる。このときのヒストグラム作成時間が最も高速であり、既存手法の約 1.5 倍の速度向上を得た。

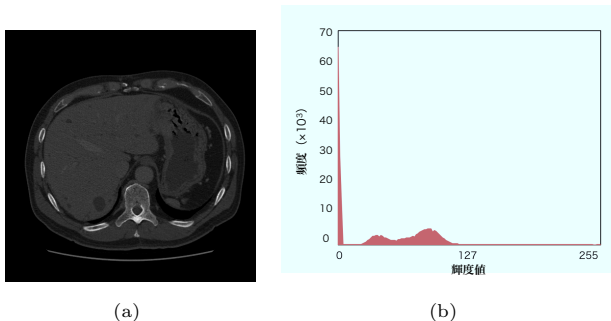


図 3 入力画像のスライスの例および輝度値の分布

表 1 実験環境

OS	Windows 7 Enterprise 64bit (SP1)
CPU	Intel Xeon X5570 2.93GHz
CUDA	v4.0 RC2
GPU	NVIDIA GeForce GTX 480
VRAM	1536 MB

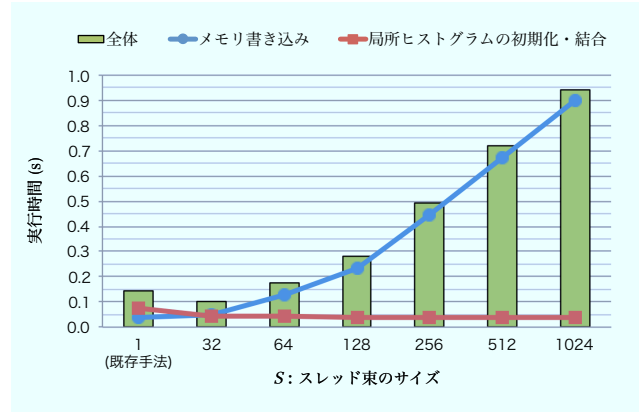


図 4 スレッド束のサイズ S ごとのヒストグラム作成時間

しかし、実効バンド幅は 8.8GB/s であり、実験で用いた GPU のピーク性能の 5% 程度である。これは、メモリ書き込み時のコアキャッシングを考慮できていないためである。したがって、今後は、メモリアクセスの効率化による高速化が課題である。

相互情報量を用いた非剛体位置合わせでは、ヒストグラムを繰り返し作成する。したがって、提案手法によるヒストグラム作成の高速化は、非剛体位置合わせの高速化において効果が期待できる。

5. まとめ

本稿では、CUDA を用いた高速なヒストグラム作成方法を提案した。局所ヒストグラムを用いたヒストグラム作成では、メモリ書き込み、および局所ヒストグラムの初期化・結合に要する時間がトレードオフの関係にある。これに着目して、提案手法は、スレッド束ごとに局所ヒストグラムを保持する。スレッド束のサイズをトレードオフ・ポイントになるように決定することにより、ヒストグラム作成を高速化する。実験の結果、特定の分布をもつ画像のヒストグラム作成で、既存手法の 1.5 倍の高速化を達成した。

今後はメモリアクセスの効率化により、さらなる高速化を図りたい。

謝辞 本研究の一部は、科学研究費補助金基盤研究 (B) (23300007) および大阪大学グローバル COE プログラム「予測医学基盤」の補助による。

参考文献

- 1) NVIDIA Corporation: NVIDIA CUDA C Programming Guide Version 4.0 (2011).
- 2) R. Shams and R.A. Kennedy: Efficient histogram algorithms for NVIDIA CUDA compatible devices, *Proc. ICSPCS '07*, pp.418–422 (2007).