I-032

GPU 上でのウェーブレット変換を用いた画像圧縮の階層的並列処理

Hierarchical Parallelization of Wavelet Transform for Image Compression on GPU

高平 真由† 吉田 明正†
Mayu Takahira Akimasa Yoshida

1 はじめに

画像圧縮手法として離散コサイン変換を用いた JPEG が広く普及しているが，さらなる高画質かつ高圧縮を実現するために，離散ウェーブレット変換を用いた画像圧縮が有効と考えられている．本稿では，Ingrid-Daubechies のウェーブレット変換 [1] を用いて，2 次元の画像に対して，行方向に 1 次元ウェーブレットを適用し，転置後にも同じく適用する．しかし，近年では画像の高精細化に伴い画素数が増加しており，画像圧縮に要する計算時間が増加しており，並列処理による高速化が期待されている．関連研究として，GPU を用いて Cohen-Daubechies-Feauveau のウェーブレット変換の高速化 [2, 3]，リフティング方式におけるメモリ使用量削減 [4] が行われているが，本手法とは方式が異なる．

本稿では，多重解像度解析を伴う Ingrid-Daubechies ウェーブレット変換の階層的並列処理手法を提案し，NVIDIA Tesla K80[5] 上で性能評価を行う．

2 離散ウェーブレット変換による画像圧縮

本章では，カラーのビットマップ画像に対して，離散ウェーブレット変換を用いた画像圧縮方法について述べる．画像圧縮は図 1 に次に示す 4 つの手順から構成される．本稿では，(2) のウェーブレット変換において，GPU による並列処理を実現する．

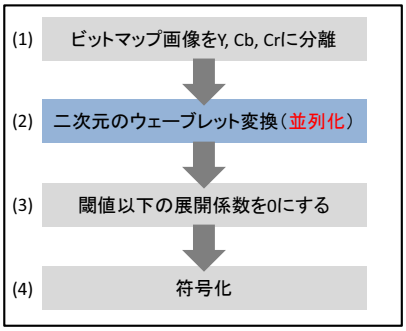

図 1 画像圧縮の手順.

2.1 離散ウェーブレット変換

離散ウェーブレット変換には，Haar のウェーブレット，Ingrid-Daubechies のウェーブレット，Cohen-Daubechies-Feauveau のウェーブレット等が存在するが，本稿では Ingrid-Daubechies のウェーブレット [1] を用いた．Ingrid-Daubechies のウェーブレットでは，予め用意された数列 p_k と q_k（N=10）を用いた．レベル $j(j >= 1)$ のスケーリング係数 $s_k^{(j)}$ と，ウェーブレット展開係数 $w_k^{(j)}$ は，式 (1) と式 (2) により求める．

†明治大学総合数理学部ネットワークデザイン学科
School of Interdisciplinary Mathematical Sciences, Meiji University

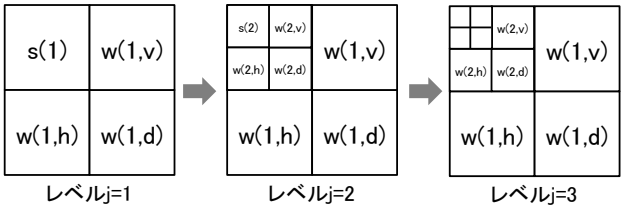

図 2 多重解像度解析.

$$s_k^{(j)} = \sum_n \overline{p_{n-2k}} s_n^{(j-1)} \quad (1)$$

$$w_k^{(j)} = \sum_n \overline{q_{n-2k}} s_n^{(j-1)} \quad (2)$$

2.2 多重解像度解析

本節では，本稿で用いた多重解像度解析の様子（レベル j=1〜3）を図 2 に示す．入力画像はレベル j=0 であり，s(0) とする．レベル j=1 において，w(1,v) には垂直方向の高周波成分，w(1,h) には水平方向の高周波数成分，w(1,d) には対角方向の高周波成分がそれぞれ現れるのが分かる．また，s(1) には s(0) を平均化した低周波成分が現れる．レベル j=2 はそれをさらに 4 分割，そしてレベル j=3 はレベル 2 のデータをさらに 4 分割した形となる．

3 GPU 上での離散ウェーブレット変換の階層的並列処理

本章では，離散ウェーブレット変換を GPU 上で階層的に並列処理する方法を述べる．

3.1 GPU 上での CUDA による並列化

提案するウェーブレット変換の階層的並列処理を GPU 上で実現するため，CUDA による並列プログラムを作成した．CUDA プログラムの各カーネル関数は，複数ブロック（MP で実行）とブロック内複数スレッド（MP 内 CUDA コアにより実行）を用いて階層的に並列処理される．本手法では，1 次元ウェーブレット変換，及び，画像データの転置をそれぞれカーネル化した．1 次元ウェーブレット変換のカーネルにおけるブロック・スレッド割当てを 3.2 節で述べる．

3.2 ウェーブレット変換のブロック・スレッド割当て

多重解像度解析の各レベル j における 2 次元ウェーブレットの過程を，図 3 の (a)，(b)，(c) に示す．まず，レベル j=0 の s(0) からレベル j=1 の (a) を求める計算では，(a) の各行を GPU のブロック (0〜B-1) に割り当て，さらに (a) の各行の要素を GPU のブロック内スレッド (0〜T-1) に割り当てる．次に，(b) の各列を GPU のブ

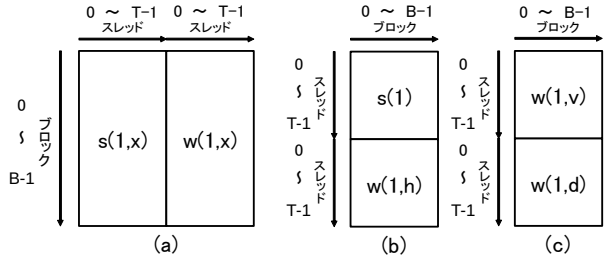

図 3　レベル j=1 のウェーブレット変換における GPU ブロック・スレッド割り当て.

ロック (0～B-1) に割り当て，さらに (b) の各列の要素を GPU のブロック内スレッド (0～T-1) に割り当てる．同様に，(c) の各列を GPU のブロック (0～B-1) に割り当て，さらに (b) の各列の要素を GPU のブロック内スレッド (0～T-1) に割り当てる．

図 3 の (a)～(c) の各計算はレベル j=1 のウェーブレット変換に対応したものであり，GPU のブロック（MP で実行）とスレッド（MP 内 CUDA コアで実行）を用いて階層的に並列処理される．

4　GPU 上での階層的並列処理の性能評価

本章では，Tesla K80 上で行ったウェーブレット変換の性能評価について述べる．

4.1　NVIDIA Tesla K80 搭載サーバ

性能評価に用いる DELL PowerEdge R730 は，GPU：NVIDIA Tesla K80，CPU：Intel Xeon E5-2680 v3 2.5GHz 12core*2，メモリ：64GB，OS：CentOS6.7，CUDA の処理系：CUDA Toolkit 7.5 となっている．

NVIDIA Tesla K80 は，Kepler アーキテクチャの GPU（GK210）を 2 基搭載した構成となっており，各 GPU は 2496CUDA コア（最大周波数 824MHz），12GB のデバイスメモリから構成されている．本性能評価では，K80 の GPU1 基を使用しており，13MP(Multiprocessor)*192CUDA コア=2496CUDA コアを用いて並列実行を行った．

なお，データのメモリ配置に関しては，GPU のデバイスメモリを使用しているが，Ingrid-Daubechies の数列 p_k と q_k は，GPU のシェアードメモリに配置している．

4.2　GPU 上でのウェーブレット変換の並列実行

本性能評価では，画像圧縮の対象となる 24 ビットカラーのビットマップ画像として，3 種類の画像ファイル（2K：2048*2048，4K：4096*4096，8K：8192*8192）を用意した．これらの画像に対して画像圧縮を行っているが，GPU 上で行った Y（輝度）成分におけるウェーブレット変換の時間のみを測定している．GPU 実行におけるブロック数は 1，16，32 とし，ブロック内のスレッド数は 256，512，1024 として実行時間を測定した．

まず，2K 画像の並列実行時間を図 4 に示す．2K 画像の 1CUDA コアによる実行時間は 80,100[ms] であり，一方，最速な並列実行時間は 16 ブロック*1024 スレッドの場合の 111[ms] であり，721 倍の速度向上が得られている．同様に，4K 画像の場合は図示していないが，1CUDA コアによる実行時間は 319,809[ms] であり，最速な並列実行時間は 32 ブロック*1024 スレッドの場合の 463[ms] であり，689 倍の速度向上が得られている．

最後に，8K 画像の並列実行時間を図 5 に示す．8K 画像の 1CUDA コアによる実行時間は 1,279,186[ms] であり，最速な並列実行時間は 16 ブロック*1024 スレッドの場合の 2384[ms] であり，536 倍の速度向上が得られている．

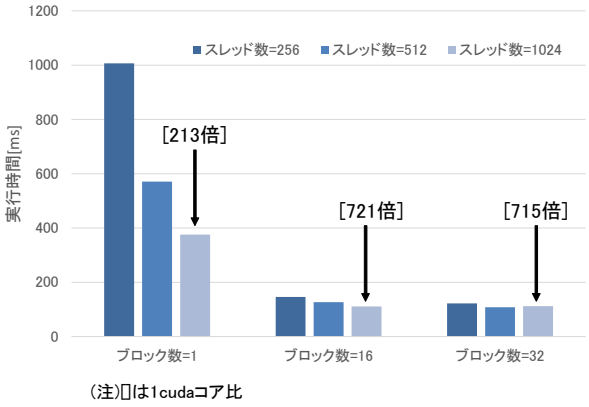

図 4　2K 画像における GPU 実行時間.

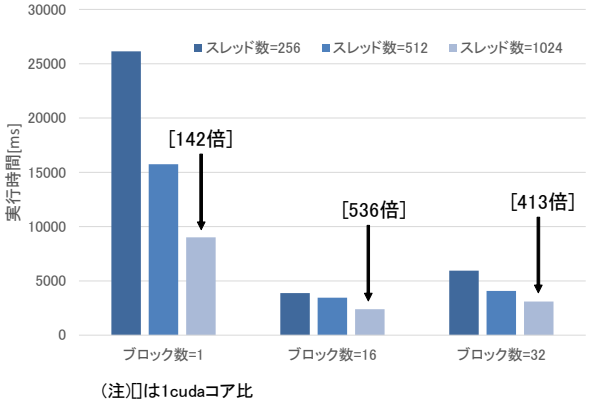

図 5　8K 画像における GPU 実行時間.

以上の結果，画像サイズに関わらず，GPU 上での階層的並列処理は，十分な速度向上を達成していることが確認された．

5　おわりに

本稿では，画像圧縮に用いられる離散ウェーブレット変換の階層的並列処理手法を提案した．多重解像度解析を伴う離散ウェーブレット変換の階層的並列処理手法を NVIDIA Tesla K80 上で実現しており，2K サイズ～8K サイズの画像に対して，十分な速度向上を得ることができた．

今後の課題としては，複数 GPU の利用と GPU メモリへのデータ配置を改善し，さらなる速度向上を達成することがあげられる．

参考文献

[1] 中野宏毅, 山本鎭男, 吉田靖夫. ウェーブレットによる信号処理と画像処理. 共立出版株式会社,1999.

[2] 上村健二, 中口俊哉, 津村徳道, 三宅洋一. GPU における離散ウェーブレット変換の速度改善. 電子情報通信学会論文誌,2007/2,Vol.J90-D,No.2.

[3] 井上昂治, 江口翔馬, 黒木祥光, 黒崎正行, 尾知博. 複数の GPU を用いたデジタルシネマ画像の実時間ウェーブレット変換. 社団法人電子情報通信学会,2011.

[4] 生澤拓也, 伊野文彦, 荻原兼一. 離散ウェーブレット変換の GPU 実装における入力の上書きによるメモリ使用量削減. 情報処理学会研究報告,2015-HPC-148-10,2015.

[5] NVIDIA. TESLA K80 GPU ACCELERATOR, https://images.nvidia.com/content/pdf/kepler/Tesla-K80-BoardSpec-07317-001-v05.pdf,2015.