GPU における離散 Wavelet 変換の速度改善

上村　健二[†]（学生員）　　中口　俊哉[†]（正員）
津村　徳道[†]（正員）　　三宅　洋一[†,††]（正員）

Acceleration of the GPU discrete wavelet transform

Kenji KAMIMURA[†], *Student Member*, Toshiya NAKAGUCHI[†], Norimichi TSUMURA[†], and Yoichi MIYAKE[†,††], *Members*

[†] 千葉大学大学院自然科学研究科, 千葉市
Graduate School of Science and Technology, Chiba 263-8522, Japan
[††] 千葉大学フロンティアメディカル工学研究開発センター, 千葉市
Research Center for Frontier Medical Engineering, Chiba 263-8522, Japan

あらまし　Wavelet 変換は画像中に局在する周波数情報を解析するための有用な手法である．Wavelet 変換を高速に実行するため GPU を計算に用いる手法が既に提案されている．本論文では GPU のパイプライン処理特性を考慮して従来手法を改良し，処理速度を 30%程高めることを可能とするアルゴリズムについて述べる．

キーワード　Graphics Processing Unit，頂点処理，離散 Wavelet 変換，GPGPU

1. ま え が き

画像中に局在する周波数情報を解析する手法として，Wavelet 変換が良く知られている．Wavelet 変換ではマザーウェーブレットと呼ばれる基底関数を拡大・縮小し，平行移動することで信号を解析するため，画像中の各位置での周波数情報を得ることができる [1], [2]．しかしながら，2 次元 Wavelet 変換は演算負荷が高いため，例えば動画像中に局在する周波数成分の実時間解析といった応用は画像サイズが大きい場合には困難であった．そのため，従来は 2 次元 Wavelet を高速に実行する必要がある場合には FPGA などのハードウェアが利用されてきた [3]．しかし，この時 PC とは別にハードウェアを別途用意する必要がある，ハードウェアの設計・制御が複雑であるなどの問題があった．

このような問題を解決するため，グラフィックス処理用のハードウェアである Graphics Processing Unit(GPU) を用いて Wavelet 変換を行う手法が提案されている．GPU は本来 3 次元仮想物体の変形や移動，画素の描画などのグラフィックス処理において，CPU の負荷を軽減する目的で導入されたものであり，当初は高性能グラフィックボードのみに搭載されていたが，近年のグラフィクス技術の発達に伴い一般的な PC への搭載も進んでいる．Hoef らは GPU の持つ描画機能に対し，フィルタリング，拡大・縮小といった処理を当てはめることにより，GPU 上で離散 Wavelet 変換を実現した [4], [5]．しかし，当時の GPU ではハードウェアベンダーが規定した機能しか利用することができなかったため，実装方法が複雑で制限も多く，かつ実行効率の面でもロスが多かった．

一方，現在の GPU では描画機能の一部をユーザがプログラム可能となり，プログラム用の高水準言語も開発された [6]．さらに内部の演算能力も飛躍的に向上し，CPU の演算能力を大幅に超えるようになった．そのため，GPU を科学技術計算など様々な一般用途に用いる研究 (General Purpose GPU:GPGPU) が盛んに行われている [7]．その一部として，Wang らは離散 Wavelet 変換をプログラマブル GPU によって実装し，高速かつ自由度の高い変換を実現した [8]．しかし，Wang らの手法は，すべての演算が 3 次元シーンの描画処理としてパイプライン処理される GPU の特性を十分考慮しておらず，さらに実行効率の改善が可能であると考えられる．そこで本論文ではこれらの特性を考慮することで，離散 Wavelet 変換をさらに高速に実行する方法を述べる．

2. GPU の処理モデル

本節では Wavelet 変換を行うために用いる GPU について簡潔に述べる．GPU を用いた 3 次元物体の描画処理の流れを図 1 に示す．描画対象のすべての頂点データはまず頂点処理により，平行移動，回転などの幾何学変換が実行される．幾何学変換された頂点は射影変換，ビューポート変換により描画対象であるデバイスの座標系へと変換される．続いてラスタライザにより，画面への描画単位であるピクセルへと分割され，各属性値(注1)が頂点間の補間処理により割り当てられる．最後に各ピクセルにおいて画面に描画される色を決定するためにピクセル処理が行われる．これらの処理は近隣の頂点・画素に対する依存関係を持たない．そのため，GPU はこれらの処理をパイプライン化し，

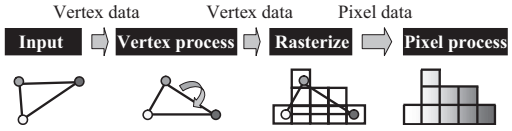

図 1　GPU による描画処理の流れ
Fig. 1　Flow of GPU rendering process.

(注1)：深度，頂点色，テクスチャ座標

並列に処理することによって高速に物体を描画することができる．また，その速度は単純に並列演算を行うパイプライン数を増加させることで向上可能であるため，動作周波数の増加よりも容易に演算速度の向上が期待できる．

近年の GPU ではこの描画処理の中で，頂点処理とピクセル処理をユーザがプログラム可能となっているため，その並列ベクトル演算能力を画像の描画だけでなく様々な科学技術計算に用いることが可能となっている．

3. プログラマブル GPU による Wavelet 変換

離散 Wavelet 変換は基本的には周波数フィルタリングで実現されるため，その処理は図 2 に示すようにフィルタカーネルと信号の畳み込みとなる．すなわち，変換画像を得るために必要な処理は各画素において，

（1） サンプル位置 (画素) を基点としたフィルタカーネルと同サイズの信号の切り出し

（2） 信号とフィルタカーネルの内積演算

である．

これらの処理は各画素において独立に適用することができる為，GPU での実装に適している．実際に GPU で扱う場合には，画像信号やフィルタカーネルは 3 次元物体に貼られるテクスチャとして定義する．

3.1 ピクセル処理のみによる離散 Wavelet 変換 (従来法)

一般に GPU の様な並列演算器では，演算処理は演算対象間 (画素間) で共通であることが要求される．したがって，離散 Wavelet 変換におけるフィルタの選択 (ハイパス，ローパス) 及び画像の端における例外処理が問題となる．Wang らはそれらを LUT テクスチャを用いて実装することで，GPU での実行を可能とした．Wang らの実装における横方向処理の場合の概略を図 3 に示す．

（1） 画素に適用するフィルタの判定

処理対象の画像は 4 つの周波数成分に分割され，それぞれ異なる位置に描画される．従って，描画対象領域では画素毎に，適応するフィルタがハイパスであるか，ローパスであるかの判定が必要となる．この判定を入力画像のサイズと分解レベルに対し事前計算し，ルックアップテーブル (LUT) テクスチャとして保存しておく．処理時には，この全画素において LUT テクスチャを参照することで共通の処理でフィルタの選択が可能となる．

（2） 画像端における例外処理

(1) で判定された結果に基づきフィルタカーネルを読み込み，画素値との内積を取る際には注目画素の近隣画素も必要となる．画像の端を超えた近隣の定義を行うためには，端を基準に折り返す，画像を繰り返すなどの例外的な画素指定が必要となる．この例外的な画素指定も，端を超えたときの画素位置の定義を事前計算し，LUT テクスチャとして保存，参照することで処理を共通としている．

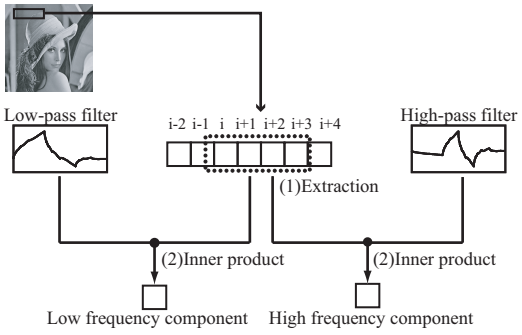

図 2 画像信号の切り出しとフィルタリング
Fig. 2 Extracting and filtering of the image signal.

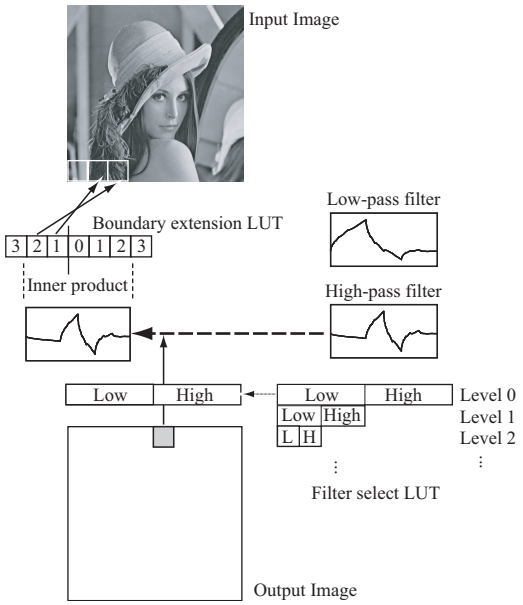

図 3 LUT テクスチャを利用した離散 Wavelet 変換の GPU への実装
Fig. 3 Impremention of discrete wavelet transform to GPU using LUT texture.

表 1　離散 Wavelet 変換の速度比較
Table 1　Speed comparison of discrete wavelet transform

	CPU		GPU	
	Pentium4 1.9	Pentium4 630	GeForce 6800	GeForce 7800 GTX
発売年	2001	2005	2004	2005
動作クロック (MHz)	1900	3000	325	430
パイプライン数	1	1	Vertex 5, Pixel 12	Vertex 8, Pixel 24
フレームレート (fps)	3.1	5.1	従来手法:22.2　**提案手法:32.2**	従来手法:47.0　**提案手法:68.0**

3.2　GPU のパイプライン処理特性を考慮した効率改善 (提案手法)

GPU 上のデータはパイプライン処理されるため，必ず頂点処理，ラスタライザ，ピクセル処理を順に通過する．このようなパイプライン処理においては各処理段階における負荷を一様にすることが望ましい．しかし，Wang らの手法では処理はピクセル処理に頼っており，頂点処理の演算資源が有効に利用されていない．そのため，ピクセル処理の中で行っている計算の一部を頂点処理とラスタライザで行うことができれば，計算負荷が分散され効率が向上することが期待できる．

提案手法における横方向処理の概略を図 4 に示す．Wang らは画像全体を一度に処理していたため，画素位置に応じて適用するフィルタを選択する処理を LUT テクスチャで実装していた．しかし，CG においては画像を複数領域に分けて描画しても結果は変わらないため，本論文では画像を低周波部分と高周波部分の 2 回に分けて描画処理する．これにより，フィルタ選択をおこなう LUT テクスチャの参照処理をピクセル処理から省くことができる．

さらに，各周波数領域において，境界処理が必要な部分を通常の領域と分けて描画し，入力画像の切り出しに用いるテクスチャ座標の計算を頂点処理にて行うようにすることで，境界処理とサンプリング座標計算もピクセル処理から取り除くことができる．これら 2 つにより演算効率を大幅に改善することができる．

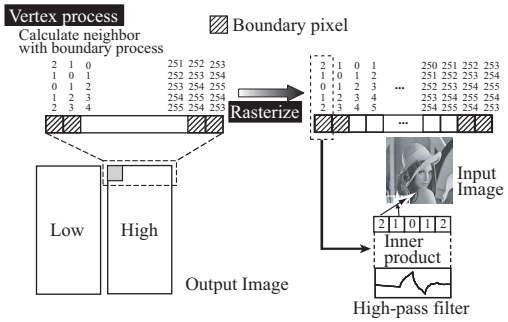

図 4　頂点処理とラスタライザを利用した負荷分散
Fig. 4　Load sharing by using vertex processing and rasterizer.

Wang らによる従来手法と，頂点処理及びラスタライザを利用した提案手法との処理の流れの比較を図 5 に示す．

4.　提案手法による高速化結果

表 1 に CPU のみによる処理，Wang らによる従来手法と提案手法の性能比較を示す．ここでフレームレートは 800×600 画素の 24bit カラー画像に対し，1 秒間に 2 次元 1 階の Wavelet 変換(注2)を行った回数を表す．

GPU を用いることにより，CPU 以上の速度で離散 Wavelet 変換が実現できている．このとき，CPU の処理速度の向上は動作クロックにほぼ線形であるのに対し，GPU ではクロック数が減少しているにもかかわらず，処理速度が大幅に向上している．これは，パイプライン数の増加によって処理が効率的に行われるためである(注3)．CPU の性能向上は一般に言われるムーアの法則に従い年間約 1.4 倍程度であるが，それに対し GPU はわずか一年で演算能力が 4 倍程度に向上しており，今後 2 次元 Wavelet 変換に GPU を用いることの有用性が増加すると考えられる．

提案手法では従来手法に比べ 30%程度の性能向上が得られ，本手法の有効性を確認することができた．また，この速度向上により一般的な動画像のフレームレートである 30fps を超える速度で Wavelet 変換を実行できるようになり，動画像に対する実時間処理も可能となった．

5.　む　す　び

GPU 特性を考慮したアルゴリズムを採用することで 2 次元 Wavelet 変換を従来より効率的に実行する手法を述べた．

(注2)：Daubecheis-9/7 による Wavelet 変換
(注3)：厳密には GPU のアーキテクチャも変更され，個々の演算効率も向上している

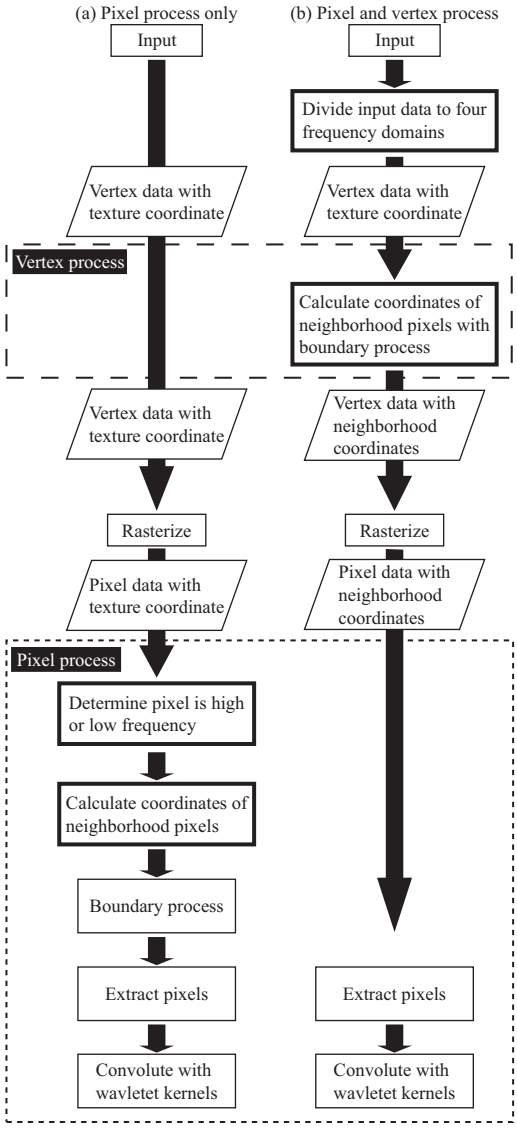

図 5　従来手法との処理の流れの比較
Fig. 5　Comparison of the process flow to conventional method.

GPU を用いることにより，2005 年の段階で CPU を用いて Wavelet 変換を実行する場合より 13 倍以上の処理速度が得られた．また，GPU の演算速度の向上は CPU と比べ早いため，今後これまで実時間処理が不可能であった高負荷な科学技術演算が，専用のハードウェアを用いずに実行可能となることが期待できる．

謝辞　本研究に多大なアドバイスを頂いた松下電器産業 (株) の本村秀人氏に感謝致します．

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, 1999.
[2] I. ドブシー (著)，山田道夫／佐々木文夫 (訳) ウェーブレット 10 講, シュプリンガー・フェアラーク東京, 2003.
[3] 仁木雅，関根優年，伊藤光，木場俊暁，”部分選択的に画像解析を行う Haar-Wavelet 変換チップ,” 電子情報通信学会技術研究報告, Vol.101, no.388, pp.97-104, 2001.
[4] M. Hoef, T. Ertl, ”Hardware Based Wavelet Transformations,” Workshop on Vision, Modeling, and Visualization, pp.317-328, 1999.
[5] M. Hoef, T. Ertl, ”Hardware accelerated Wavelet Transformations,” VisSym, pp.93-103, 2000.
[6] W. Mark, R. Glanville, K. Akeley and M. Kilgard, ”Cg: A System for Programming Graphics Hardware in a C-like Language,” ACM Transactions on Graphics, 22, 3, pp.896-907, 2003.
[7] J. Owens, D. Luebke, N Govindaraju, M. Harris, J. Krüger, A. Lefohn and T. Purcell, ”A Survey of General-Purpose Computation on Graphics Hardware,” Eurographics 2005, State of the Art Reports, pp.21-51, 2005.
[8] J. Wang, T. T. Wong, P. A. Heug and C. S. Leung, ”Discrete Wavelet Transform on GPU,” Proceedings of ACM Workshop on General Purpose Computing on Graphics Processors, pp. C-41, 2004.

（平成 xx 年 xx 月 xx 日受付）

Abstract Wavelet transform is a powerful technique to analyze local spatial frequency of an image. In this paper, we improve the speed of the GPU discrete wavelet transform by using not only pixel processing but also vertex processing. This approach achieve load-sharing between steps of the rendering process. Practical evaluation for performance showed the improvement of calculation speed about 30%.

Key words Graphics Processing Unit, Vertex Processing, Wavelet transform, GPGPU