3K-1

GPUを用いた大規模計算機ホログラム生成プログラムの最適化

宮田 裕章 † Boaz Jessie Jackin †† 大川 猛 † 大津 金光 † 横田 隆史 †
早崎 芳夫 †† 谷田貝 豊彦 †† 馬場 敬信 ††
† 宇都宮大学大学院工学研究科情報システム科学専攻 †† 宇都宮大学オプティクス教育研究センター

1 はじめに

計算機ホログラム（CGH : Computer Generated Holography）は次世代の3Dディスプレイを実現するための有望な手段として期待されている．しかし, 3次元像を表示するデバイス技術の開発とともに, 非常に大きな計算コストがネックになっており, CGHを用いた3Dディスプレイは実用化されていない．

本稿では大規模なオブジェクトを対象として, 計算コストの削減についてオブジェクト分割法を用いることにより, リアルタイムでの大規模なホログラム生成における並列化の方法を提案する．同時に, 並列処理にGPU（Graphics Processing Unit）を用いることで, 計算の高速化を図る．

2 大規模CGH生成のオブジェクト分割法による計算

CGHは, 再生したい像のもととなるオブジェクトデータにフーリエ変換を行うことで得ることができる．CGHの1つの画素に対して, オブジェクトの全ての画素からの寄与の計算を行うため, その計算量は膨大になる．しかし, CGHの各画素の計算はオブジェクトデータのみに依存しており, 互いの計算は独立している．そのため並列計算を行うことで大きな速度向上が期待できる．CGHの計算はGPUなどの並列計算を得意とする演算装置を用いることで効率的に計算することができる．本稿においては, GPUを用いてCGH計算の高速化を図る．

CGHの計算をGPUで行う場合, オブジェクトのデータを一度CPUからGPUへと転送する必要がある．オブジェクトデータが1つのGPUに収まる容量であれば, オブジェクトデータ全体に対してフーリエ変換を行うことで容易にCGHを得ることができる．しかし, オブジェクトデータが1つのGPUに収まらないほど大規模になると, オブジェクトを分割し, GPUに収まるデータ量ずつ計算しなければならない．この場合, GPUクラスタなどの複数のGPUを用いた環境ならば, 分割された各オブジェクト（サブオブジェクト）を異なるGPUで計算することで高速化が図れる．しかし, CGHの各ピクセルは全オブジェクトデータに依存しているため, オブジェクトを分割した場合は各GPUでの計算結果を参照するために, GPU間での通信が頻繁に発生する．この通信はリアルタイムでのCGH生成の大きな妨げとなるため, 大規模CGHの計算に通常のフーリエ変換を用いる手法は不向きである．

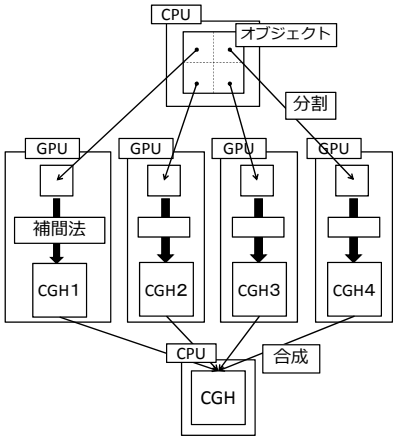

図 1: オブジェクト分割と複数GPUによるCGH計算

我々はこの問題に対して, オブジェクトを分割してCGHを計算する方法を提案している [1, 2, 3]．この手法によりサブオブジェクトからその領域のみの情報を持ったCGH（サブCGH）を生成することが可能になった．この方法において, 各サブオブジェクトからサブCGHを生成する計算は互いに独立している．そのため, GPU間の相互通信なしに複数のGPUで各サブオブジェクトを並列に計算することができる．また, サブCGHは足し合わせることで, 全ての領域の情報を持ったCGHにすることができる．（図1）

3 CGH計算におけるコアレッシング

GPUを用いるにあたり, 本研究ではCUDAを使用する．プログラミングの際は, 使用しているGPUアーキテクチャやプログラムのアルゴリズムに則ったチューニングを行わないと, GPUの性能を十分に発揮できない．本稿では, その中で重要なメモリアクセス効率の改善について述べる．

本研究では2次元画像に対してオブジェクト分割法を適用した．図2にオブジェクト分割法における処理のフローを示す．オブジェクト分割法は行列の行方向, 列方向を分けて処理する．行方向の処理を行う場合は, 配列の行方向, つまり連続した領域へのアクセスとなり, 複数のデータがまとめてアクセスされるコアレッシングが発生する．コアレッシングアクセスになると, 複数のデータに並列にアクセスできるため, メモリア

Speedup of Large-scale Computer-Generated-Hologram Program Using GPU

†Hiroaki Miyata, ††Boaz Jessie Jackin, †Takeshi Ohkawa, †Kanemitsu Ootsu, †Takashi Yokota, ††Yoshio Hayasaki,††Toyohiko Yatagai and ††Takanobu Baba
Department of Information Systems Science, Utsunomiya University (†)
Center for Optical Research and Education, Utsunomiya University (††)

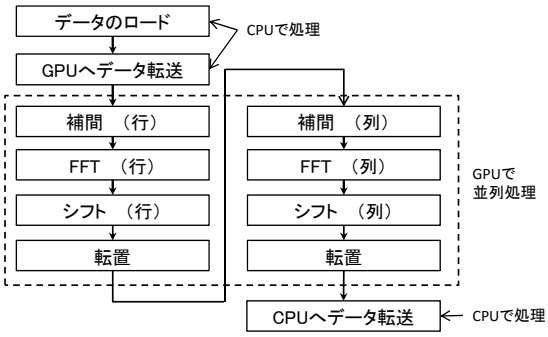

図 2: オブジェクト分割法のフロー

クセスの効率が上がる．しかし, 配列の列方向の処理を行う場合, 列方向のアクセスは飛び飛びの領域へのアクセスとなりコアレッシングが発生しない．各データは逐次的にアクセスされるためメモリアクセスの効率は大幅に低下する．

表 1 左より, 行方向と比較して列方向は処理時間が大幅に増加していることが分かる．列方向は行方向と比べ, 処理量が 2 倍にはなるが, 行方向と同様の処理ができていれば 5ms 前後で完了するはずである．プログラムのプロファイルを行った結果からもコアレッシングアクセスがされていないことが確認できた．

そこで, 行列の転置を行い, 列と行を入れ替えることで, 列方向のアクセスもコアレッシングされるように改善した．転置はシェアードメモリを用いて行う．シェアードメモリに行列データの一部を読み出し, 行と列を入れ替えてストアする．(図 3)　シェアードメモリ上ではバンクコンフリクトを回避すれば, 列方向でも並列にアクセスすることができる．

4　実行性能

実行環境は CPU:Intel core i7 920 (2.67GHz), GPU: NVIDIA Tesla C1060 (1.30GHz) コア数 240, オブジェクトデータのサイズは 4096 × 4096, オブジェクトの分割数は 4(2 × 2) である．

表 1 に最適化前と最適化後の各関数の処理時間を示す．最適化後の結果を見ると, コアレッシングするようにメモリアクセスを改善したあとは補間 (列) とシフト (列) の実行時間が大幅に短縮されていることが分かる．本来は無かった転置の処理を加えたことによる実行時間の増加はあるが, 全体として実行時間が短縮

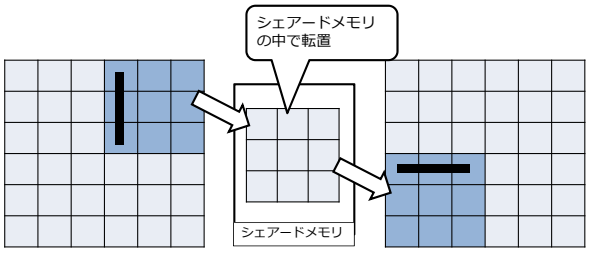

図 3: シェアードメモリを使った転置

表 1: 各関数の処理時間 (単位：ms)

関数	最適化前	最適化後
補間 (行)	2.23	2.38
FFT（行）	2.69	3.17
シフト（行）	2.37	3.04
転置	-	2.65
補間（列）	26.95	4.96
FFT（列）	5.32	5.65
シフト（列）	24.36	6.16
転置	-	4.94
計	63.92	32.95

され, 最適化前と比較して 1.94 倍の速度向上となった．オブジェクト分割法における計算部分のみの実行時間は 30ms 程度だが, これに CPU と GPU 間の転送に掛かる時間を加えると約 65ms 程度の時間がかかる．リアルタイムでの CGH 生成による動画像作成には 30〜60fps 程度のフレームレートが必要になると仮定すると, CGH 1 枚あたりの計算時間は約 30ms 以下に抑える必要がある．このことから, 今回実験で使用した以上の大規模な CGH 計算の際は, プログラムのさらなる最適化に加え, データ転送に掛かる時間の改善も必要になると考えられる．

5　おわりに

本稿ではオブジェクト分割法による CGH 生成を GPU を用いて高速化した．メモリアクセスを改善することでプログラムの最適化を行った．最適化の結果 1.94 倍の速度向上が得られた．今後はオブジェクトの分割方法を変更することによる処理削減を試み, さらなる高速化を図る．

謝辞

本研究は，一部日本学術振興会科学研究費補助金 (基盤研究 (C)24500055，同 (C)24500054, 同 (C)25330055, 若手研究 (B)25730026) の援助による．

参考文献

[1] Boaz Jessie Jackin, et al.: “Proposal of Fast Calculation for Large-Scale Fresnel Hologram using Interpolation Method,”, International Workshop on Holography and Related Technologies 2013 (IWH 2013), 15d-4 Oct, 2013.

[2] Takanobu Baba, et al.: “INTERPOLATION-BASED OBJECT DECOMPOSITION AND PARALLEL COMPUTATION METHOD FOR LARGE-SCALE COMPUTER-GENERATED HOLOFRAM,”, Parallel and Distributed Computing and Networks 2014 (PDCN 2014), Feb, 2014.(accepted)

[3] 宮田 裕章, Boaz Jessie Jackin, 他.: “補間法による大規模計算機ホログラム生成の GPU による高速化,”, Optics & Photonics Japan 2013 (OPJ 2013), 13pPD5 Nov, 2013.