5H-04

大規模計算機ホログラム生成プログラムのマルチGPUを用いた高速化

渡邉　晋平 † 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 ††
† 宇都宮大学工学部情報工学科　†† 宇都宮大学オプティクス教育研究センター

1　はじめに

3D ディスプレイを実現する方法の一つとして計算機ホログラム (CGH) がある．CGH によって我々は視野角によらず，専用のフィルタを必要とすることなく 3D 画像を見ることが出来る．しかし，この CGH の実現には課題がある．まず，25cm 大の再生像で人間の目に見えるような可視域を作り出すためには 14.7G ピクセルの CGH が必要とされることが分かっている．この CGH の各画素を float 型の複素数で表すと 117GB という非常に大きなデータサイズとなる．このようなデータサイズを処理するためには大きな計算コストがかかる．また動画として画像を人の目に見せるためには 30fps 以上で画像を連続表示する必要がある．そのためリアルタイムで CGH を計算し表示するには CGH の生成に速度が求められる．

本稿ではオブジェクト分割法 [1] を用いて大規模なデータに対応するとともに，マルチ GPU (Graphics Processing Unit) を用いて並列処理をすることで CGH 生成の高速化を図る．

2　マルチ GPU を用いたオブジェクト分割法

CGH は入力となるオブジェクトにフーリエ変換をすることで求めることが出来る．GPU を用いて計算する場合，オブジェクトのデータサイズが 1 つの GPU のデバイスメモリに収まる大きさならば 2D-FFT を用いて容易に CGH を得ることができるが，それを越える場合はオブジェクトデータを一つの GPU のデバイスメモリに収まるサイズに分割し処理する必要がある．その方法の一つとして 1D-FFT と転置を用いた方法 (TS 法) がある．

TS 法では分割したオブジェクトデータ (サブオブジェクト) をそれぞれ GPU に転送し 1D-FFT を行い，CPU 側にデータを戻して転置するという処理を 2 度繰り返して CGH を得る方法である．この方法ではマルチ GPU を用いることでサブオブジェクトに対して 1D-FFT を並列に行える．しかし転置を行う際にデータを CPU 側にまとめる必要があるので GPU 間での同期が必要となる．そこで，オブジェクト分割法を用いる．

オブジェクト分割法は分割した各オブジェクトからその分割した領域の情報のみを持った CGH (サブ CGH) を生成し，それらを足し合わせること (重ね合わせ) によって CGH を得る手法である．この方法では各 GPU 間の同期を必要とせずサブオブジェクトの計算を図 1 のように並列に行うことが出来る．

なお，CGH を再生する際に光学的に再生機で連続してサブ CGH を切り替えて再生することで人間の目には一つの CGH として捉えさせることが出来る．この方法を用いれば重ね合わせの処理を省略することが出来る [2].

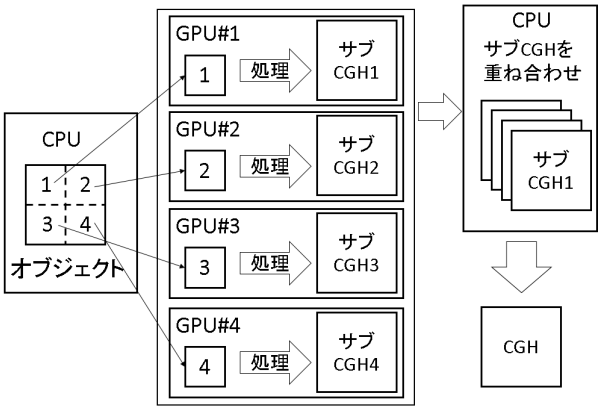

図 1: マルチ GPU を用いたオブジェクト分割法

3　オブジェクト分割法の改善

プログラムを高速化するために GPU にあった調整をする必要がある．本稿では分割方法の変更とストリーム処理を用いてプログラムの高速化を図る．

3.1　分割方法の変更

一般的に GPU 側から CPU 側のメモリを参照することは出来ないので GPU を用いるプログラムでは CPU 側から GPU 側へのデータの転送が必要となる．従来のオブジェクト分割法では図 2 の上の図のように縦方向，横方向にそれぞれ分割をしているが，この分割法では分割した各サブオブジェクトのデータはメモリ上の配置で不連続になる．CPU と GPU 間のデータ転送では一度にメモリ上の不連続な領域にあるデータを送ることが出来ないため，CPU 側でデータを連続領域に配置し直すか複数回に分けてデータを転送する必要がある．ここで図 2 の下の図のように分割方向を縦方向にのみにすることで分割したデータをメモリ上で連続な領域にすることができる．この変更により CPU で

Acceleration of large scale computer generated hologram calculations using multi-GPU
†Shinpei Watanabe, ††Boaz Jessie Jackin, †Takeshi Ohkawa, †Kanemitsu Ootsu, †Takashi Yokota, ††Yoshio Hayasaki, ††Toyohiko Yatagai and ††Takanobu Baba
Department of Information Science, Faculty of Engineering, Utsunomiya University (†)
Center for Optical Research and Education, Utsunomiya University (††)

のオブジェクトデータの再配置処理が不要になり，再配置用に確保していたメモリ領域の削減ができた．

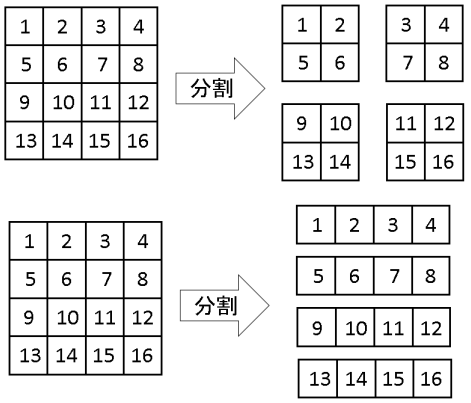

図 2: 分割方法の変更

3.2 ストリーム処理

サブ CGH を求めるために使用するメモリ量が GPU のデバイスメモリを越えないようにするために，オブジェクト分割法による処理では一度にサブ CGH 全てを求めるのではなく複数回に分けてサブ CGH の一部を求めて CPU 側に転送する処理を繰り返す．図 3 の上の図に処理の流れを示す．サブ CGH の一部を求める処理は互いに依存関係がないため GPU のストリーム処理機能を用いれば図 3 の下の図のようにサブ CGH の一部の計算とすでに計算が完了したサブ CGH の CPU へのデータ転送を同時に行える．ストリーム処理実装前では重ね合わせ・GPU 制御を一つの CPU スレッドで行っていたが，ストリーム処理実装後では重ね合わせと GPU 制御のスレッドを分けることで GPU 制御をしながら重ね合わせができる．

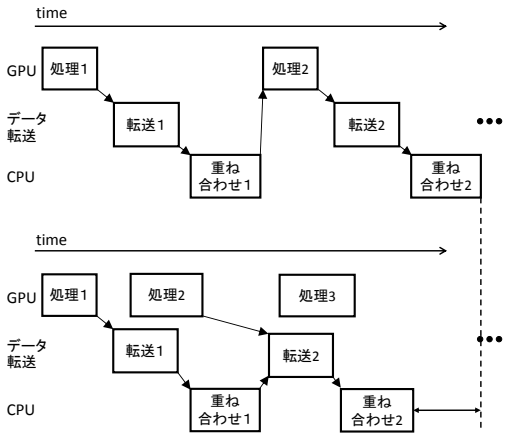

図 3: オブジェクト分割法による CGH 生成の流れ

4 実行時間の評価

3 章で述べた変更を加えたオブジェクト分割法の評価を行う．実行環境は CPU: Intel Xeon X5650 (2.67GHz) コア数 6，GPU: NVIDIA Tesla M2090 (1.3GHz) コア数 512，使用した GPU の数は 4，オブジェクトの分割数は 4 である．

図 4 にプログラムの実行時間のグラフを示す．I は従来の分割方法でストリーム処理をしていないプログラム，II は分割方法の変更を行ったプログラム，III は分割方法の変更とストリーム処理を行ったプログラムであり，それぞれの実行時間を示している．サイズはオブジェクトの画素数を表している．全ての結果において全体の実行時間では II のプログラムが最も短かった．III のプログラムでは転送の時間を隠蔽できたが，重ね合わせ処理にかかる時間が増え II と比べて長くなった．重ね合わせ処理の際に CPU 側のスレッドを生成することにより CPU 側の合計のスレッド数が増え，物理コアの数を越えるのが原因と考えられる．重ね合わせ処理を省略した場合には 9216 と 16384 のサイズにおいて III が最も速い．

I と II を比べると最大で 1.23 倍の速度向上となった．重ね合わせ処理を省略した場合には I と III を比べると最大で 1.29 倍の速度向上となった．

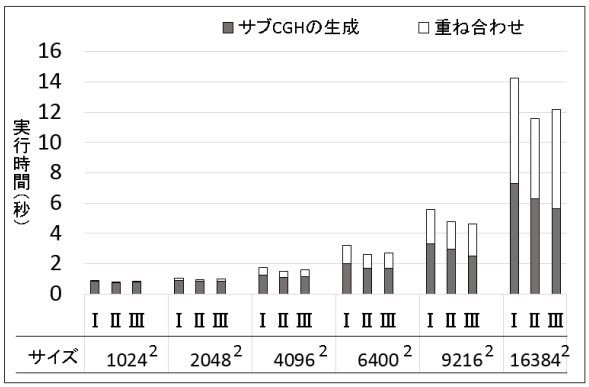

図 4: プログラムの改善による実行時間の変化

5 おわりに

本稿ではオブジェクト分割法を用いた CGH 生成プログラムをマルチ GPU を用いて高速化した．分割方法の変更で 1.23 倍の速度向上が得られた．今後はストリーム処理の際に時間が増えた重ね合わせ処理の時間の削減を試み，更なる高速化を図る必要がある．

謝辞

本研究は，一部 JSPS 科研費 (C)24500055，同 (C)15K00068，同 (C)25330055 の援助による．

参考文献

[1] 宮田 裕章，他.：“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”，情報処理学会 第 76 回全国大会講演論文集，pp. 1-191 〜 1-192，2014.

[2] Boaz Jessie Jackin, et al.: “Distributed calculation method for large-pixel-number holograms by decomposition of object and hologram planes”, Optics Letters, Vol. 39, No. 24, pp. 6867-6870, 2014.