B-023

GPU を用いた電子ホログラフィ専用計算機の構築

Comparison of PTX codes written in different parallel programming languages

魚谷　智志 *　祢津　佑 *　吉田　慧吾 *　増田　信之 *
Satoshi Uotani　Tasuku Netsu　Shogo Yoshida　Nobuyuki Masuda

1. まえがき

近年，3 次元テレビや 3 次元顕微鏡など，ホログラフィ技術をコンピュータに応用した研究が行われている．3 次元テレビの実用化には様々な課題があり，ソフトウェア面においてはホログラムの作成に膨大な時間がかかる点が挙げられる．今日では，膨大な計算時間のかかる問題に対して，GPGPU がよく用いられる．GPU は，計算コアが多く並列処理が得意であるため，数値計算に応用することで計算の高速化が期待できる．本研究ではホログラムの作成を高速に行うために，GPU を用いた電子ホログラフィ専用計算システムの構築を目的とした．

2. ホログラフィ

ホログラフィとは，光の干渉や回折を利用して 3 次元像の記録・再生が行える光技術のことである．図 1 に 3 次元像記録の原理を示す．3 次元像を記録する場合，光源であるレーザ光をビームスプリッタで 2 分割し，一方を物体に当てた物体光，もう一方をそのまま参照光とする．物体光と参照光を記録媒体に照射することで，記録面上に位相のずれによる干渉縞 (ホログラム) が記録される．本研究では，ホログラムの記録をシミュレーション上で行う CGH(Computer Generated Hologram) 計算について高速化を行った．

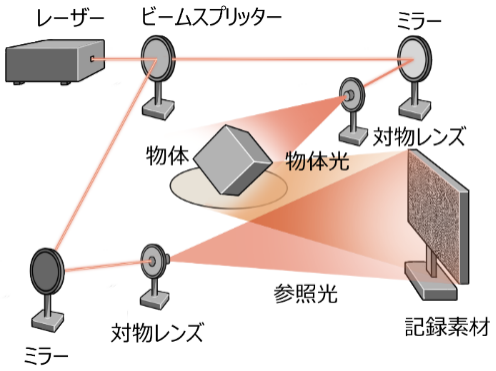

図 1: 3 次元像の記録原理

2.1 CGH

CGH とは，電子ホログラフィにおけるホログラムを記録する工程のことである．

ここで，図 2 に示した点光源が 1 点のホログラムについて考える．点光源の分布は，球面波として考えることができ，条件として点光源の振幅 A は 1，初期位相 ϕ は 0 となるため物体光は，

$$\begin{aligned} O(x,y) &= \frac{A}{r}\exp(i(kr+\phi)) \\ &= \frac{1}{r}\exp(ikr) \end{aligned} \tag{1}$$

と表される．参照光についても同様に，振幅は 1，初期位相は 0 となり，

$$\begin{aligned} R(x,y) &= A_R(x,y)\exp(i\phi_R(x,y)) \\ &= 1\times\exp(i0) = 1 \end{aligned} \tag{2}$$

と表される．物体光と参照光はホログラム面で重なり合うため光強度は，

$$\begin{aligned} I(x,y) &= |O(x,y)+R(x,y)|^2 = \left|\frac{1}{r}\exp(ikr)+1\right|^2 \\ &= \frac{1}{r^2}+1+\frac{1}{r}\exp(ikr)+\frac{1}{r}\exp(-ikr) \end{aligned} \tag{3}$$

となる．第 1 項と第 2 項は，それぞれ物体光と参照光の直接光であり 3 次元再生に寄与しないため，CGH の計算処理には不要となる．また，式 (4) において，係数 2 は全体にかかるだけで本質的意味合いは無いため，

$$\begin{aligned} I(x,y) &= \frac{1}{r}\exp(ikr)+\frac{1}{r}\exp(-ikr) = \frac{2}{r}\cos(kr) \\ &= \frac{1}{r}\cos(kr) \end{aligned} \tag{4}$$

を求めれば良い．

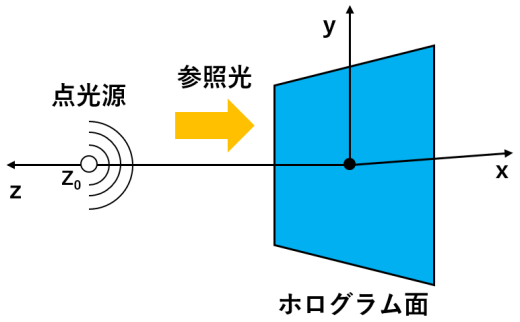

図 2: 点光源とホログラム面の位置関係

3. シミュレーション方法

本シミュレーションでは，初めに物体点データをホスト PC から GPU に転送する．続いて，GPU 上で CGH 計算が行われ，計算結果をホスト側に転送する．最後に，ホスト PC で計算結果から画像データを作成する．今回は，この一連の流れを 16 回行うと共に GPU 上で行う CGH 計算と，データ転送について高速化を行った．

*東京理科大学基礎工学部

3.1 最適化手法

本シミュレーションでは，3 つの最適化手法を用いた．1 つ目は CGH 計算式の最適化である．本シミュレーションで作成されるホログラムは 2 値化されるため，近似計算を用いても計算結果に影響はない．したがって，開平計算を取り除くフレネル近似や，三角関数の近似を行った．また，定数部分をホスト PC で計算し，除算を取り除くことで計算コストの削減を行った．

2 つ目はメモリアクセスの改善である．図 3 に GPU のプログラマブルメモリの概略図を示す．メモリアクセスの改善においては，Global メモリへのアクセスの削減を行った．まず，Shared メモリに書き込めるだけ物体点データを書き込み，CGH 計算を行う際は Shared メモリから物体点データにアクセスを行うようにした．また，各物体点における計算結果をレジスタに格納するようにした．この一連の流れを全物体点の CGH 計算に対し行い，全ての光強度計算を終えた後に Global メモリに計算結果を書き込むことで，Global メモリへのアクセスの削減を行った．

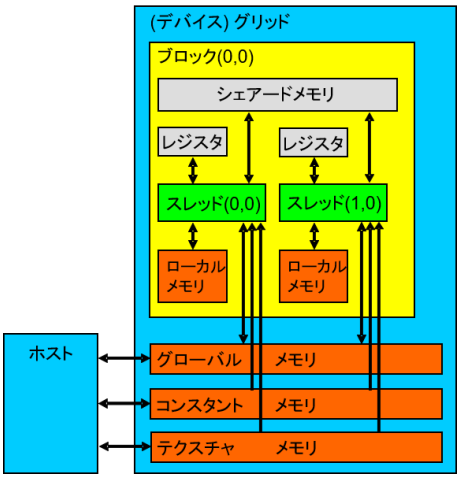

図 3: GPU のプログラマブルメモリの概略図

3 つ目はデータ通信時間の隠蔽である．GPU では Stream を用いることでカーネルの実行とデータ通信を同時に行うことができる．そこで，データ通信とカーネル実行を Stream 単位で管理し，Stream を複数立ち上げることでカーネルの実行とデータ通信をオーバーラップさせ，データ通信時間の隠蔽を行った．

3.2 シミュレーション環境

シミュレーション環境を表 1 に示す．本シミュレーションでは float 型で計算を行い，16 枚のホログラムを作成した．物体点数 N を 11646，画素数を 2048×1024，thread 数を x，y 両方向共に 16 とし，参照光の波長 λ を 633 nm，画素ピッチ p を 10.5 μm とした．計算時間の計測には gettimeofday() 関数を使用し，計測範囲は 1 枚目の CGH 計算に用いる物体点の転送開始時間から，16 枚目の CGH 計算の計算結果がホスト PC に転送されるまでとし，計測時間は 50 回の平均を算出した．

表 1: 開発環境

CPU	Intel Core i7 8700K (3.70 GHz)
RAM	64 GB (DDR4-2400)
GPU	NVIDIA GeForce GTX 1080Ti
OS	Ubuntu 18.04
compiler	CUDA10.0

3.3 計測結果

本シミュレーションの計測結果を表 2，作成したホログラムを図 4 に示す．表 2 より，CPU 用プログラムを単純に GPU 用に実装した場合，1 枚辺りの計算時間は約 1.9 秒かかり，フレーム数換算すると 0.52 FPS であったのに対し，最適化を行うことで動画として知覚可能な 24.27 FPS となった．また，高速化比を見るとシンプル実装より 46.6 倍高速に動作した．ただし，本シミュレーションにおけるフレーム数換算は，画像データを作成する工程は含んでいない．そこで，画像データ作成にかかる時間について計測を行ったところ，処理時間は約 11.910 ms であった．

表 2: シンプル実装と最適化実装の計算時間

	シンプル実装	最適化実装
計算時間 [ms]	30732.8	659.274
計算時間 (1 枚)[ms]	1920.8	41.205
フレーム数 [FPS]	0.52	24.27
高速化比	1	46.6

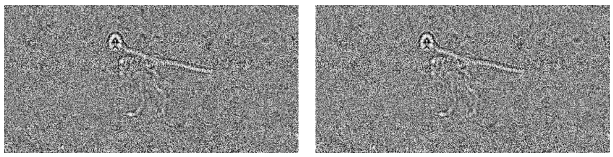

図 4: シンプル実装 (左) と最適化実装 (右) のホログラム

4. まとめと今後の課題

本稿では，GPU を用いて CGH 計算の高速化を行い，物体点数 11646 点の 3D データを用いて 2048×1024 ピクセルのホログラムを作成した結果，フレーム数換算で 24.27 FPS となり，シンプル実装と比較して約 46.6 倍高速に計算を行うことができた．

しかし，本シミュレーションでは，高級言語レベルでのチューニングしか行っていない．そこで，疑似アセンブリ言語である PTX の解析を行い，直接 PTX を書き換えることで中間言語レベルでのチューニングを行い，更なる高速化を目指したいと考えている．

参考文献

[1] 伊藤智義，下馬場朋禄 著，『ホログラフィ入門 コンピュータを利用した 3 次元映像・3 次元計測』，講談社，p171，2017.