D-101

GPU を用いた空間分割によるリアルタイムレイトレーシングの検討

Real -time Raytracing using Space Partitioning on GPUs

岡崎大輔 †　　孟林 ‡　　山崎勝弘 ‡

Daisuke Okazaki　Meng Lin　Katsuhiro Yamazaki

１．はじめに

レイトレーシング法では、光線と物体の交差判定に最も多くの処理時間を必要とする。本研究の目標は、GPU を用いてレイトレーシングの処理時間を 100ms 以下にすることである。そのために、GPU 上で画面分割と空間分割を組み合わせて高速化を図る。画面分割では、画面をインターリーブ分割し、各ブロックをストリーミングマルチプロセッサ（SM）に割り当てることにより、各画素の計算を並列処理する。空間分割では、等空間分割と適応型空間分割について検討する。光線が通過する空間内の物体のみと交差判定を行うことにより、交差判定の回数を減少させ、処理の高速化を図る。

２．ＧＰＵの構成

本研究では、GeforceGTX480 を使用する。図１に GPU の構成を示す。GeforceGTX480 はギガスレッドスケジューラと 4 個 GPC から構成されている。図２に SM の構成を示す。各 GPC には 4 個の SM 及び、ラスターエンジンが搭載されている。各 SM 内に 32 個のストリーミングプロセッサ(SP)を持ち、正弦関数、余弦関数を計算する。SFU(Special Function Unit)を 4 個、ワープスケジューラとディスパッチユニットをそれぞれ２個持つ。また、共有メモリ、レジスタファイル、ロードストアユニットを持つ。

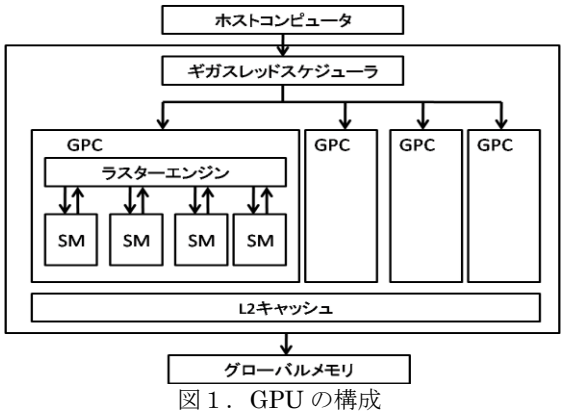

図１．GPU の構成

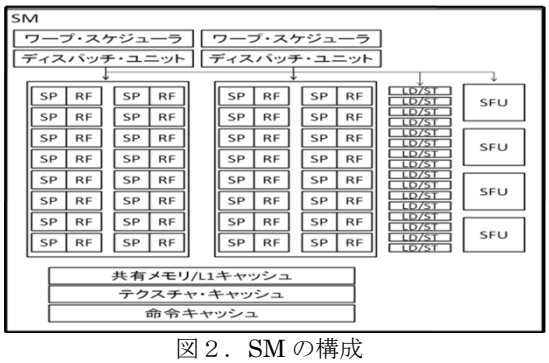

図２．SM の構成

３．空間分割の検討

レイトレーシング法では、光線と物体の交差判定に最も多くの処理時間を必要とする。そこで物体との交差判定の回数を減らすために空間分割を行う。空間分割とは、物体が存在する空間を登録し、それ以外の空間とは交差判定を行わなくすることである。空間分割には等空間分割、適応型空間分割がある。また、空間分割ではボクセルで空間を管理しているため、光線がどのボクセルを通過するか知る必要がある。これを知るためにデジタル微分解析法(3DDDA)を用いる。

３．１　等空間分割と適応型空間分割

等空間分割とは、空間をボクセルとよばれる一定の大きさに分割し、その各空間に存在する物体の情報を得る。次にレイトレーシングの処理を行うが、空間分割では視線が入るボクセルに属している物体とのみ交差判定を行う。

適応型空間分割では、あらかじめ物体の分布を調べ、物体数の多いボクセルをより細かく分割する。これにより、物体数を均等にすることで、ほぼ同じ交差判定の回数にすることで効率化を図る。適応型空間分割の各ボクセルは、等空間分割のように単純に決定できないため、サブボクセルを用いて決定する。サブボクセルとは、物体の存在する空間を一定の大きさに分割したものであり、最初にそれらのサブボクセルに対して属する物体を登録し、次に物体の存在する空間を２つに分割する。２つに分割された空間のうち物体数の多い空間をさらに２つに分割する。適応型空間分割のボクセルの決定を図３に示す。

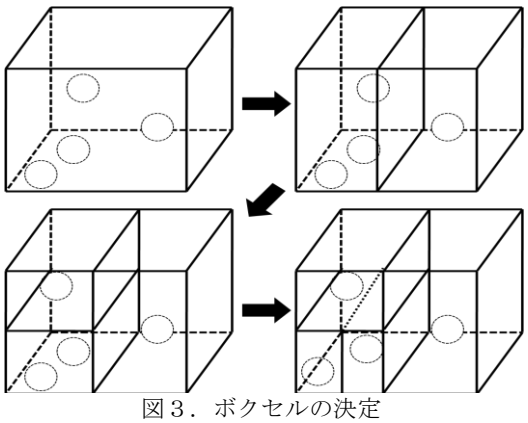

図３．ボクセルの決定

３．２　３ＤＤＤＡによる光線の移動

デジタル微分解析法を用いることにより光線が次に入るボクセルを知ることができる。どのボクセルに光線が入るかは、現在のボクセル番号(i,j)において、次のボクセルの縦軸(i+1)、次のボクセルの横軸(j+1)との交点での直線の長さを比較し、直線の長さの短い方に+1 または－1 した空間へ次の光線が入ることがわかる。3DDDA によるボクセルの移動を図４示す。図４の例では、次のボクセルの縦軸(i+1)

† 立命館大学大学院理工学研究科,Graduate School of Science and Engineering , Ritsumeikan University
‡ 立命館大学理工学部,College of Science and Engineering , Ritsumeikan University

との交点までの直線の距離を tx、次のボクセルの横軸(j+1)との交点までの直線の距離を ty とする。
①ボクセル(0,0)の場合、tx=点[1~2]の距離、ty=点[1~3]の距離となり、tx<ty なので次の空間は(i+1.j)で(1,0)となる。
②ボクセル(1,0)の場合、tx=点[2~4]の距離、ty=点[2~3]の距離となり、ty<tx なので次の空間は(i.j+1)で(1,1)となる。

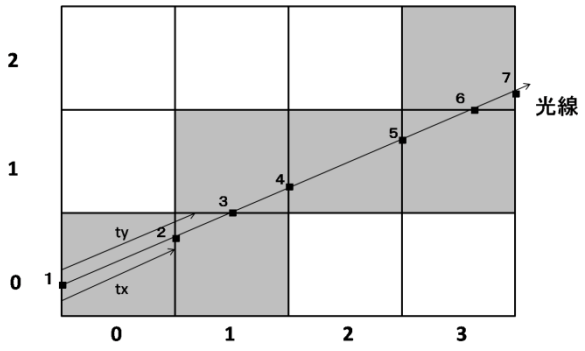

図４．３ＤＤＤＡによる光線の移動

４．ＧＰＵを用いた画面分割と空間分割の実装

本研究では GPU を用いてスクリーンの各画素について並列化を行い、同時に物体が存在する空間を認識するために、空間分割を適用する。画面分割により１画素に対して SM 内の SP で光線を追跡し、交差判定、反射、輝度計算を行う。画面を分割することで、光線探索の並列化を図り、それぞれの光線に対し、空間分割を適用することで、交差判定の回数を減らし高速化を図る。画面分割と空間分割のプログラムの流れを図５に示す。

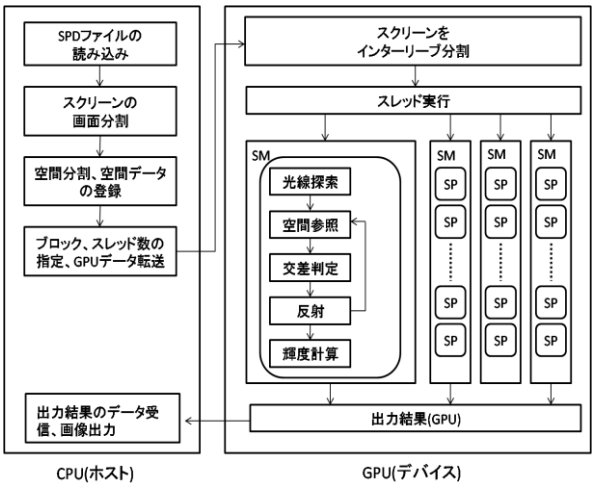

図５．プログラムの流れ

４．１　画面分割

レイトレーシング法では、画面上のある画素の輝度値を求める処理が、他の画素の輝度値を求める処理に依存していないため、並列処理の効果を得やすい。画面分割の手法はインターリーブ分割である。スクリーンが 512*512 の解像度の場合、ブロック数 64*64、スレッド数 8*8 が最速であることが分っている[3]。図６に示すようにまずスクリーンを 64*64 に２次元インターリーブ分割する。SM0 は行方向で 0,4,8..番目、列方向で 0,4,8..番目を担当する。各ブロックは 8*8 の画素があるので、32 個の SP それぞれが、１画素の処理を担当し、交差判定、反射、輝度計算を行う。SM は 16 個あるので、インターリーブ分割で各ブロックを割り当てる。

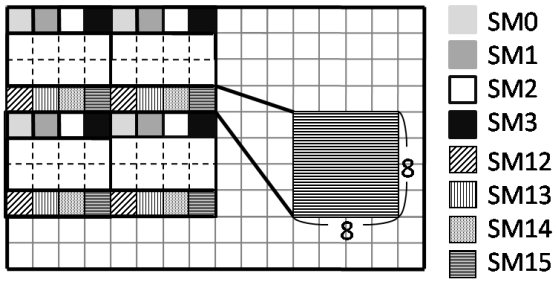

図６．スクリーンの２次元インターリーブ分割

４．２　空間分割

空間分割のアルゴリズムとして、最初に CPU 上でボクセル情報を調べ、ボクセル情報データを登録する必要がある。その後、GPU 上のグローバルメモリに空間情報データを転送する。分割された空間情報データを SP から高速にアクセスが可能な共有メモリに記憶させる。各 SP の処理について、どのボクセルに光線が入るかどうかを判別する必要があり、3DDDA を用いて判別する。光線が入ったボクセル内に物体がある場合、その物体との交差判定を行い、反射した光線の探索を行う。もし物体が存在しない場合は、現在のボクセルを抜け、次にどのボクセルに光線が入るかをボクセルデータを参照し判断する。この、ボクセル移動、交差判定を、空間全体から光線が抜けるまで繰り返し行う。

５．おわりに

本論文では、GPU を用いたリアルタイムレイトレーシングの画面分割と空間分割による並列化について検討した。レイトレーシングの各画素の輝度値を求める処理が、他の画素の計算と独立であることを用いて、スクリーンをインタリーブ分割し、各ブロックを各ストリーミングマルチプロセッサで並列実行する。

今後、GPU 内での処理方式をさらに詳細に検討し、処理方式を実現して、画面分割と空間分割による高速化を評価する予定である。

参考文献

[1]上野謙二郎：GPU を用いたリアルタイムレイトレーシングの並列化の検討と実現,立命館大学大学院,理工学研究科修士論文,2011
[2]上野謙二郎,孟林,山崎勝弘：GPU を用いたリアルタイムレイトレーシングの検討,情報処理学会第 74 回全国大会,1ZB-1,2012
[3]孟林,上野謙二郎,山崎勝弘：GPU を用いたリアルタイムレイトレーシングの並列化,第 12 回情報科学技術フォーラム,FIT2012,2C-1,2012
[4]吉谷崇史：空間分割による並列レイトレーシング法,立命館大学大学院,理工学研究科修士論文,1997