社団法人　情報処理学会　研究報告
IPSJ SIG Technical Reports

2005－ＳＬＤＭ－121（13）
2005／10／20

動画生成用レイトレーシングエンジン

帰山 芳行† 鈴木 健一† 中村 維男†

†東北大学大学院情報科学研究科 〒980-8579 仙台市青葉区荒巻字青葉 6-6-01

E-mail: †{yoshi,suzuki,nakamura}@archi.is.tohoku.ac.jp

あらまし 高品質なコンピュータグラフィックスを生成することが可能なレイトレーシング法を用いて、物体の移動を伴う動画像を生成する要求が存在する。しかし、このような動画像の生成においては、フレームごとに空間内における物体の位置情報の更新処理が発生するため、従来の静止画像生成を目的としたレイトレーシングエンジンでは処理の高速化が困難であるという問題があった。本論文では、この問題を解決するための物体並列モデルに基づく動画像生成レイトレーシングエンジンを提案する。さらに、特に処理要素間の負荷分散の問題を考察する。

キーワード コンピュータグラフィックス，レイトレーシング，並列処理，負荷分散

A Ray-Tracing Engine for Dynamic Scene Rendering

Yoshiyuki KAERIYAMA† Kenichi SUZUKI† and Tadao NAKAMURA†

†Graduate School of Information Sciences, Tohoku University. Aramaki Aza Aoba 6-6-01, Aoba-ku, Sendai-shi, 980-8579 Japan

E-mail: †{yoshi,suzuki,nakamura}@archi.is.tohoku.ac.jp

Abstract There exists a strong demand for high quality computer graphics by using ray-tracing for dynamic scene where objects are moving. However, due to the reason of overhead in reconstructing the object-space information of all the frame updates, it is difficult to speed up the dynamic scene rendering by using a dedicated hardware for static scene rendering. In order to solve this problem, we proposed a new ray-tracing engine that enables dynamic scene rendering, which is based on object parallel processing model. In this report, we investigate the problems of load balancing between processing units for the hardware that had been proposed.

Keyword Computer Graphics, Ray-Tracing, Parallel Processing, Load Balancing

1. はじめに

近年、工業製品や建築物のデザイン検討においてコンピュータグラフィックスの利用が進んでおり、これらの分野では高品質な画像の高速生成技術の確立が強く望まれている。現在は、ハードウェア技術と要求品質のトレードオフから、ラスタ処理と呼ばれる物体座標系からスクリーン座標系への位置変換とフラグメント処理を組み合わせた画像生成手法を用いることが一般的である。またこれらの演算には、ベクトルデータの繰り返し処理を高速に演算する GPU(Graphics Processing Unit)と呼ばれる演算処理要素を搭載した画像生成専用ハードウェアが利用される。GPU を用いたラスタ処理は、高速なアニメーション生成などを得意とする反面、光の物理現象の再現を要する画像生成は困難であり、上記に求められる品質を満たしきれていない。

一方レイトレーシング法は、光源から出る光線が物体と相互作用を行いながら最終的に視点に集まる原理とその可逆性を利用し、視点から発する光線と物体の交点を求めてその物体を表示する手法である。これは、光と物体の相互作用を忠実に再現することにより、ラスタ処理では再現できない反射や屈折といった現象の表現を可能とする高品質画像生成の最も基本的手法である。しかし本手法は、視点から追跡する光線と定義空間に存在する全ての物体との交差判定の処理に膨大な計算量を要するため、動画像生成への適用は困難とされてきた。

本報告では、画像生成手法として高品質画像の生成が可能なレイトレーシング法を用いつつ、物体の移動を伴う動画像を生成可能な画像生成エンジンを提案する。提案するハードウェアは、物体並列処理モデルを用いてレイトレーシング処理を並列化し、処理の高速化を目指す。特に、交差判定処理要素に注目し、演算処理要素間の負荷分散手法について議論を行う。

2. 動画像生成のためのレイトレーシング

2.1. 空間分割法を用いたレイトレーシング処理

レイトレーシング法は、大域照明モデルと呼ばれる照明効果を計算するための基本的処理である。例えば、このレイトレーシング法を用いて 1 枚の静止画像を生成する場合、図 1 に示すように 3 次元物体を投影する

スクリーンの画素ごとに視点に入射する光線を逆追跡し、画素に入射する輝度値を求めることで画像生成を実現する。レイトレーシング法を用いた動画像は、このようにして求めた静止画像を連続して表示することで得られる。本手法は、光線の伝播現象を物理的に忠実にモデル化した写実性の高い画像生成手法といわれているが、反面3次元空間内に位置する全ての物体と、画素から発する全ての光線との交差判定処理が必要となるため、その膨大な処理量が問題となる。

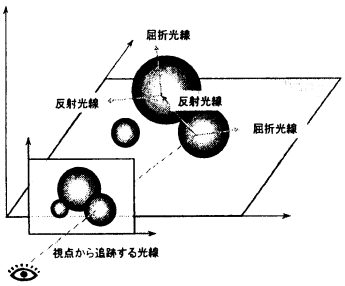


図1. レイトレーシング法の概念図

レイトレーシング処理において、全ての光線と全ての物体の交差判定処理を行うことは、現実的ではない。このため、交差判定処理を効率良く行ういくつかの手法が提案されている。その中でも、空間分割手法は効果の高い手法とされている。これは、図2に示すように物体定義空間内をいくつかの小空間に分割し、光線が通過する小空間内に存在する物体とのみ交差判定を行うことにより、交差判定処理量を大幅に削減する手法である。

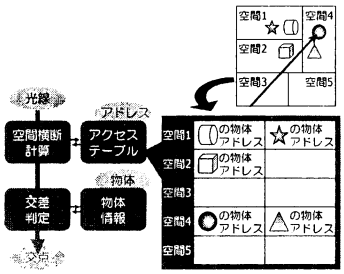


図2. 空間分割による交差判定処理の削減

本手法は、画像生成を開始する前段階の処理として、分割された小空間に位置する物体情報のポインタを階層化して空間テーブルに格納する必要があるが、画像生成時の処理時間を短縮することができるため、静止画像の生成や物体の移動を伴わず観察者の視点のみが変化する動画像(ウォークスルー画像)の生成に用いられてきた。しかし、本研究が対象とするような物体の移動を伴う画像生成処理においては、動画像を構成する1枚1枚の静止画像を生成するたびに、再び空間を分割し空間テーブルを作成しなければならない。このため、動画像生成に適した空間テーブルの再作成を必要としない交差判定処理量の低減手法が求められる。

2.2. 物体の移動を扱うレイトレーシング処理

本節では、本研究で取り扱う物体情報の定義について述べる。物体は、複数のプリミティブから構成されているものとする。物体の移動とは、物体を構成するプリミティブが、その相対位置を変化させることなく平行移動や回転を伴って位置を変化させるものであるとする。個々のプリミティブを表現する幾何情報は、物体ごとのローカル座標系を用いて記述されているものとする。これにより、ローカル座標系とワールド座標系のアフィン変換行列のみを更新していくことで物体の移動を表現することが可能となり、各物体のプリミティブに関する幾何情報を更新することなく、移動する物体を表現することが可能となる。同時に、動画像生成を開始する直前に各物体定義空間に対して空間分割処理を施し、結果得られる空間テーブルをフレームごとに更新することなく用いることが可能となる。

上記のように定義した物体に対し、動画像レイトレーシング処理は以下のように実現する。まず画像生成開始前に、物体を構成するプリミティブ全てを包含する包含領域の形状を求めておく。次に、光線と全ての包含領域との交差判定処理を行う。包含領域内と交差した光線のみ、交差した領域のローカル座標系へのアフィン変換を行う。ローカル座標系内においては、従来の空間分割を用いた画像生成と同様に光線と物体の交差判定処理を行う。全ての交差物体の中から、始点からの距離が最小となる物体を最近傍物体として得ることができる。このような処理手順により、各フレームを生成する前に空間テーブルを再作成することなく、処理量の増加を最小限としながら高速な動画像生成を実現することが可能である。

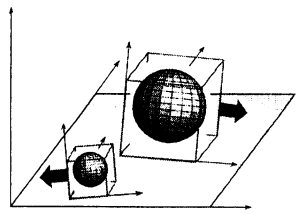


図3. 物体と包含領域

3. 提案するハードウェアの構成

3.1. 物体並列処理による高速化

レイトレーシング処理の高速化のためには、計算手法の工夫だけでなく、並列化及びメモリバンド幅問題

の解決が必須である。レイトレーシング処理中の光線と物体の様々な処理には、光線ごとまたは物体ごとに空間的な並列性が内在する。これらの並列性を効率よく利用することができるハードウェア構成が求められる。また、膨大な量の光線情報および物体情報を演算処理要素にフェッチするために発生するメモリバンド幅問題が、避けて通れない問題であり、これを解決する処理の局所化が求められる。

本報告で提案するレイトレーシングエンジンは、物体ごとに処理を並列化する物体並列処理モデルに着目する。本並列処理モデルは、物体情報を格納するメモリ近傍に演算処理要素を配し、各々の処理要素に光線情報をブロードキャストすることにより、処理の分散と局所化を図り、メモリバンド幅の問題を解決する並列処理モデルである。

3.2. ハードウェアの構成

提案するハードウェアは図 4 に示すように複数の処理要素から構成される。汎用プロセッサ (General Purpose Processor ,GPP)は、光線の生成やシェーディング計算などを担当する。このような計算は、処理の流れが複雑であり、またアプリケーションごとに処理内容が変化するため、複雑な制御の記述が可能でかつ高いユーザプログラマビリティを有する汎用プロセッサコアが必要となる。複数の交差処理ユニット(Intersection Unit. IU)は、包含領域との交差判定や包含領域内に存在するプリミティブとの交差判定の処理を行う。

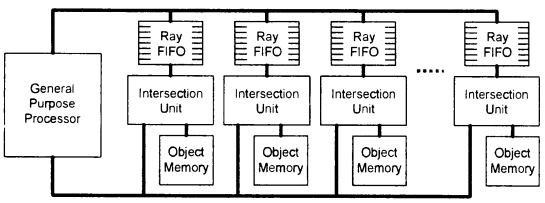


図 4. ハードウェアの構成

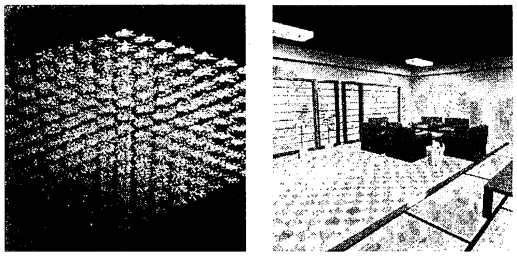
3.3. 処理の流れ

前節で述べたハードウェアによる動画像生成の処理の流れについて述べる。まず、動画像を生成する前段階の処理として、物体の幾何情報を各々の物体メモリ(Object Memory)に格納する。動画像の 1 フレームを構成する静止画像の生成を開始する前に、GPP において包含領域の位置やローカル座標へのアフィン変換を記述する行列を計算し、各々の IU に送信する。画像生成が開始されると、GPP は視点情報から光線情報を生成し、各処理要素の光線 FIFO に分配する。各 IU は、FIFO から光線情報を受け取り、包含領域との交差判定処理を行い、交差していれば光線のローカル座標系への座標変換の後、その包含領域内のプリミティブとの交差判定処理を行う。光線と交差したプリミティブの情報は、IU から GPP に送信されシェーディングの計算などに用いられる。

包含領域の形状には直方体や球など様々な形状があることが知られているが、提案するハードウェアでは球(バウンディングスフィア)を採用することとする。その理由は、直方体を採用した場合と比較し、光線と包含形状との交差判定に必要な計算量・データ量が最も少ないからである。

4. 性能評価

提案するハードウェアは、物体並列処理モデルを採用することで、光線と物体の交差判定処理の分散と局所化を目指している。しかし、本ハードウェアによる交差判定処理は、最も処理量の多い演算処理要素の処理時間が全体の処理時間となってしまうため、各処理要素間の負荷のばらつきが重要な要素となる。そのために本章では、任意の静止画像をレンダリングする処理に注目し、提案するハードウェアの各 IU の処理量を調査し、負荷のばらつきを確認することとする。



(a)polygons (b)room

図 5. 2 種類のテストシーン

テストシーンとしては、図 5 に示す 2 種類のシーンを用いる。シーン(a)polygons は、動画像生成ベンチマーク BART[9]の Museum シーンにおいてシーン中央を舞う三角形ポリゴン群を 3 次元空間中に格子状に配置したものである。物体プリミティブは全て三角形パッチでパッチ数は 65536 枚である。また、物体を包含する包含領域数は 32 個である。シーン(b)room は、部屋の内部を表現したテストシーンである。物体プリミティブは全て三角形パッチであり、その数は 35108 枚である。包含領域数は 32 個である。

生成する画像は 256x256 画素であるとし、初期光線は 1 画素に 1 本生成される。シェーディングは Phong シェーディングを行うと仮定し、IU に分配する光線は初期光線と影光線の 2 種類である。なお、反射や屈折光は扱わない。

IU は、図 6 に示すよう加減算器 1 個と乗算器 1 個およびレジスタファイルから構成されるものとする。加

減算および乗算はともに1サイクルで終了すると仮定し、このとき IU は交差判定およびアフィン変換命令を表1に示すサイクル数で処理を完了するものとする。今回の調査では、空間分割処理は行われないものとし、包含領域に突入した光線は領域内の物体と総当り判定を行うものとする。その他のハードウェアは理想的であると仮定する。GPP は十分に高速で、各 IU は GPP の処理を待つことはない。また、光線 FIFO のあふれや、バスへのアクセス競合も無いものとする。

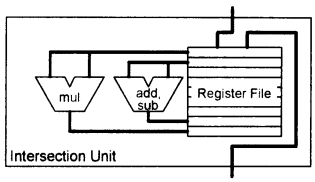
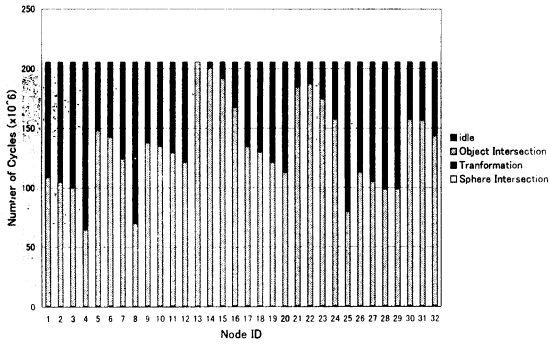
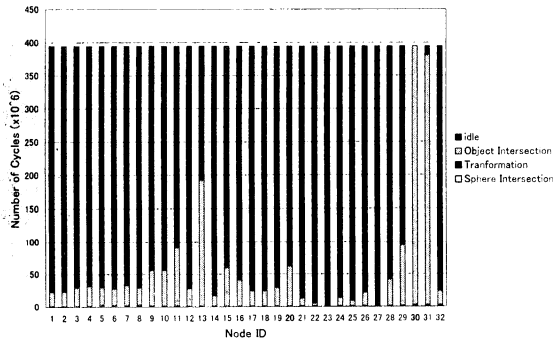


図 6.交差ユニットの詳細



(a)polygons



(b)room

図 7. 各処理の実行サイクル数

表 1.光線と球の交差判定に必要な処理サイクル数

球交差判定	9 サイクル
座標変換	19 サイクル
物体交差判定	4～22 サイクル

図 7 は、各 IU ノードにおける球交差判定・座標変換・物体交差判定・アイドルの各処理サイクル数を示したものである。包含領域の大きさほぼ同じでかつ領域内の物体数が等しいシーン(a)では、空間中に物体が偏らずに分散していることもあり、IU 間の処理が分散されていることがわかる。しかし、実際のレンダリング環境に近いシーン(b)では、特定の IU に処理が集中してしまい他のノードのアイドル時間が極端に大きくなってしまうことがわかった。

5. 負荷分散に関する検討

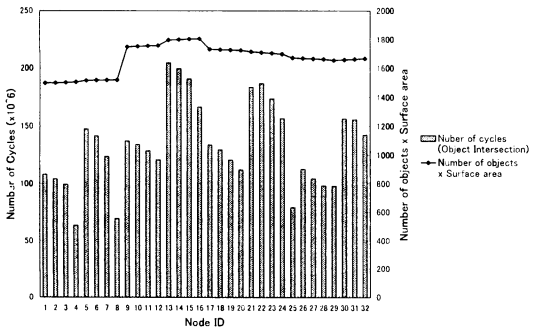
5.1. 球の表面積と物体プリミティブ数との相関

前章の性能評価からわかるように、IU ごとの交差判定処理に不均衡が発生する可能性があり、これを均一化する負荷分散手法の適用が求められる。各処理要素における交差判定処理量は、領域内のプリミティブ数と各 IU が担当する領域を通過する光線の量によって決まる。つまり、この値を IU ごとに一定にすることで、IU 間の処理の不均衡を最小限にすることが可能となる。

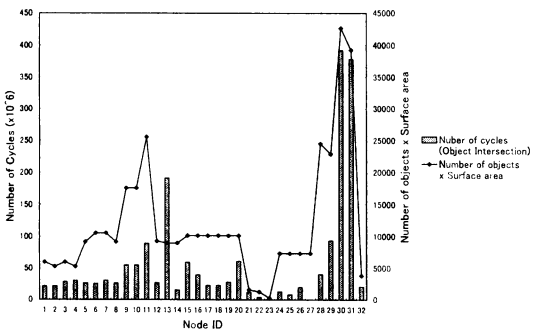
包含領域を通過する光線の量は、シーンごとに視点の位置やシェーディングの条件などで左右される値であるため、動画像を生成する前に求めることは不可能である。しかし、各 IU が処理を担当する包含領域の表面積と物体プリミティブ数の積の値を一定とすることが、確率的に包含領域を通過する光線量を一定に近づけることであると考えられる。

図 8 には、2 種類のベンチマークシーンにおける包含領域の表面積と物体プリミティブの数の積の値と、交差判定処理に要したサイクル数の関係を示した。シーン(b)room では、積の値に大きなばらつきがあるため、処理要素間の交差判定処理量に大きな不均衡を発生させていたことがわかる。一方シーン(a)polygons は、包含領域表面積と物体プリミティブ数の積がほぼ一定となるようなシーンであるため、シーン(b)と比較し交差判定処理量が均一であることがわかる。

しかし、物体数と包含領域表面積の積がほぼ一定であるシーン(b)においても、並列処理効率はそれほど高いとは言い難い。これは、視点によりスクリーンに投影される面積が多い包含領域ほど、通過する光線の数が集中してしまうために発生する処理の不均衡であると考えられる。このため、上記負荷分散手法に加えて、包含領域を通過する光線数が視点情報に大きく依存しない負荷分散手法が求められる。



(a)polygons



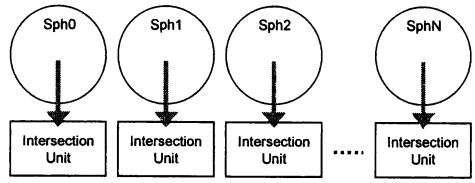
(b)room

図 8. 交差判定処理量と境界表面積・物体数の積との相関

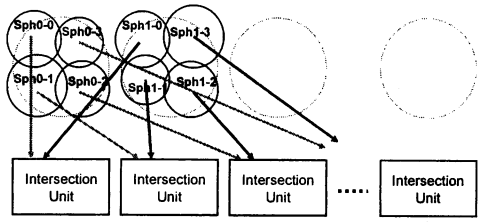
5.2. 包含領域の細分割

前章の性能評価では、同一方向へ移動する物体を 1 つのバウンディングスフィアで包含し、その包含領域に関する処理を 1 つの IU に割り当てる方式とした。しかし、ユーザが記述する 3 次元物体は、個々の物体を構成するプリミティブ数は一定であるという保障はなく、また包含領域の大きさもまちまちであるという問題があり、シーン(b)においては並列処理効率が著しく低下してしまった。さらに、シーン(a)のように、たとえプリミティブ数や包含領域の大きさを一定に近かったとしても、視点により負荷の不均衡が発生する可能性があることがわかった。

この問題の解決のひとつのアプローチとして、図 9 に示すようなバウンディングスフィアの細分割手法を検討する。1 つの物体を構成するプリミティブを複数の球を用いて包含しなおし、再作成した球に関する処理を異なる処理要素に割り当てる。包含領域を細分割することで、球との交差判定処理やローカル座標系への光線情報のアフィン変換は、細分割数分増加する可能性があるが、包含領域の表面積と物体プリミティブ数の積の値を IU ごとに均一化することができるため、また視点による処理の偏りのリスクを低減することができるため、並列処理効率を向上させることができると考えられる。

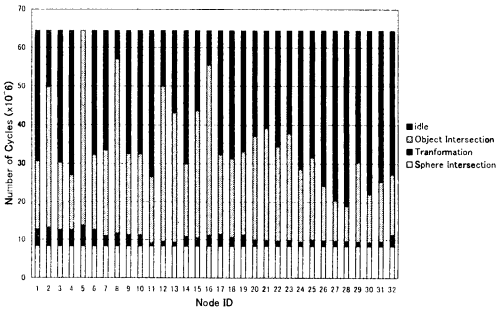


(a) 1 つの球を 1 つの IU に割り当て

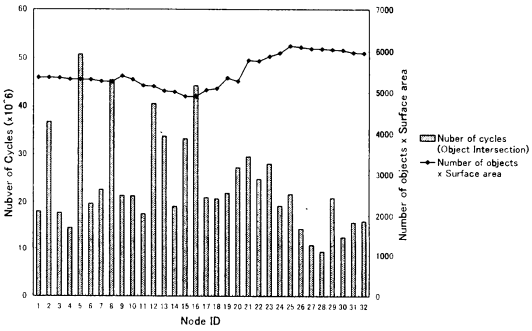


(b)1 つの球を分割し複数の IU に割り当て

図 9. バウンディングスフィアと処理の割り当て



(a)各処理の実行サイクル数



(b)交差判定処理と境界領域表面積及び物体数の積との相関

図 10. 包含領域細分割の効果

本節では、前章の実験で負荷の不均衡が大きいシーン(b)について、1つの球をさらに8個に細分割し、包含領域の表面積と物体プリミティブ数の積の値が処理要素ごとに均一となるよう IU への処理の割り当てを行い、IU の負荷の均衡について調べた。その結果、図10 に示すように演算器が停止しているアイドル時間は大幅に低減することができ、並列処理効率の改善を確認できた。

6. 関連研究

レイトレーシング法のハードウェア処理による高速化に関する研究は、これまで様々なものが提案されている。ラスタ処理を専門に行う GPU の高速な SIMD 演算能力に注目し、レイトレーシング処理を実装し高速な画像生成を行う試みも行われている [1] [2] [3]。しかし搭載メモリ量の制限や、GPU 特有の複雑なメモリアクセスを必要とする問題から、実用に見合った処理速度を得られてはいない。

文献[8]で提案される 3DCGiRAM は、物体定義空間を分割して得られた小空間内の処理を、仮想的に3次元に配置されたメモリに格納し、これらのメモリに画像生成のための演算処理要素を統合した機能メモリである。しかし、ウォークスルー動画に特化した本アーキテクチャを用いて移動する物体を扱うことは困難である。文献[7]は、本報告と同様に交差判定を包含領域と物体の2種類にわけて処理を行う手法を用いて、レイトレーシング処理を FPGA に実装したものであるが、物体の移動に関する扱いは考慮していない。

Slusallek らは、文献[4]で動的物体のレイトレーシング処理方式として BSP 木に構造化した空間情報をフレームごとに更新する方法を提案している。また文献[5][6]において、[4]で提案する処理方式を、FPGA を用いたプロトタイプシステムに実装を行っているが、フレームごとの空間分割情報の更新処理は未だ実装されていない。

これに対して、本報告で提案する動画像生成用レイトレーシングエンジンは、フレームごとの空間分割処理を行わない効率の良い動画像生成手法を、並列処理により高速に実現するハードウェアであると考えられる。

7. まとめ

本報告では、物体の移動を考慮した動画像生成専用レイトレーシングエンジンの提案を行った。提案する手法は、空間分割情報を再作成することなく、包含領域との交差判定処理および光線情報の座標変換処理を行うことで、移動する物体のレイトレーシング処理を実現する。本手法の交差判定処理を、物体並列処理モデルに基づいて並列化し、高速に処理することが可能なハードウェア構成について述べた。さらに処理要素間の負荷の偏りについて調査を行い、負荷の不均衡を削減するひとつの手法を示した。

今後の課題として、包含領域内部におけるプリミティブとの交差判定処理に空間分割手法を導入して負荷の偏りを再評価すること、現在はひとつの IU に実装している包含領域との交差判定・座標変換・プリミティブとの交差判定処理を、各々独立した処理要素に割り当てるハードウェア構成の検討、などがあげられる。

文　献

[1] Timothy J. Purcell, Ian Buck, William R. Mark, and Pat Hanrahan, "Ray Tracing on Programmable Graphics Hardware" ACM Transactions on Graphics. 21 (3), pp. 703-712, 2002. (Proceedings of ACM SIGGRAPH 2002).

[2] Timothy J Purcell, Craig Donner, Mike Cammarano, Henrik Wann Jensen, Pat Hanrahan, "Photon Mapping on Programmable Graphics Hardware" Proceedings of the ACM SIGGRAPH/ EUROGRAPHICS Conference on Graphics Hardware, pp. 41-50, 2003.

[3] Nathan A. Carr and Jesse D. Hall and John C. Hart, "The Ray Engine" Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware. pp. 37-46, 2002.

[4] Ingo Wald ,Carsten Benthin and Philipp Slusallek, "Distributed Interactive Ray Tracing of Dynamic Scenes" PVG '03: Proceedings of the 2003 IEEE Symposium on Parallel and Large-Data Visualization and Graphics. 2003.

[5] Jorg Schmittler, Sven Woop, Daniel Wagner, Wolfgang J. Paul and Philipp Slusallek, "Realtime Ray Tracing of Dynamic Scenes on an FPGA Chip" Proceedings of the ACM SIGGRAPH/ EUROGRAPHICS conference on Graphics hardware. pp. 95- 106, 2005.

[6] Sven Woop ,Jorg Schmittler and Philipp Slusallek, "RPU: a programmable ray processing unit for realtime ray tracing" Proceedings of ACM SIGGRAPH 2005. pp. 434 - 444, 2005.

[7] Joshua Fender , Jonathan Rose, "A High-Speed Ray Tracing Engine Built on a Field-Programmable System" IEEE ICFPT(International Conference on Field-Programmable Technology) 2003. pp. 188 - 195, 2003.

[8] Hiroaki Kobayashi, Ken-ichi Suzuki, Kentaro Sano, Yoshiyuki Kaeriyama, Yasumasa Saida, Nobuyuki Oba, Tadao Nakamura, "3DCGiRAM: An Intelligent Memory Architecture for Photo-Realistic Image Synthesis" ICCD '01: Proceedings of the International Conference on Computer Design: VLSI in Computers & Processors (ICCD'01). 2001

[9] Jonas Lext, Ulf Assarsson, and Tomas Moeller. BART:A benchmark for animated ray tracing. Technical report, Department of Computer Engineering, Chalmers University of Technology, Goeteborg, Sweden, May 2000. Available at http://www.ce.chalmers.se/BART/.