УДК 004.925.3

Д. К. Боголепов, Д. П. Сопин, Д. Я. Ульянов, В. Е. Турлапов

ПОСТРОЕНИЕ SAH BVH ДЕРЕВЬЕВ ДЛЯ ТРАССИРОВКИ ЛУЧЕЙ С ИСПОЛЬЗОВАНИЕМ ГРАФИЧЕСКИХ ПРОЦЕССОРОВ

Разработан алгоритм построения SAH BVH деревьев на графических процессорах в режиме реального времени. Алгоритм позволяет более чем на порядок увеличить производительность визуализации по сравнению с аналогами.

Ключевые слова: *трассировка лучей, ускоряющие структуры, SAH, BVH, динамические сцены, графический процессор, OpenCL.*

Метод трассировки лучей традиционно используется в компьютерной графике для синтеза изображений высокого качества. Первые реализации этого метода на графических процессорных устройствах (ГПУ) допускали обработку сцен только со *статической* геометрией (см., например, [1]).

В настоящей работе рассматривается подход к визуализации *динамических* сцен с использованием ускоряющей структуры в форме деревьев BVH. Данная структура обеспечивает наиболее „плотную“ аппроксимацию геометрии сцены при минимальном числе узлов; в процессе построения дерева используются только *центроиды* треугольников, что исключает случай пересечения треугольником плоскости разбиения. Производительность процесса трассировки лучей напрямую зависит от *качества* построения дерева, наилучшим критерием которого является *эвристика площадей поверхности* (SAH) [2]. На данный момент, по-видимому, отсутствуют эффективные реализации алгоритмов построения SAH BVH деревьев на ГПУ, которые обеспечили бы возможность визуализации произвольных динамических сцен [3].

Целью настоящей статьи является разработка эффективного алгоритма построения SAH BVH деревьев на ГПУ в режиме *реального* времени. Предполагается, что все объекты сцены представлены набором треугольников (*triangle soup*), при этом допускаются произвольные изменения геометрии во время визуализации. Приведем общую схему алгоритма построения SAH BVH дерева.

1. На каждом шаге выполняются все доступные задания. В результате возможно наиболее полно использовать ресурсы ГПУ, а также снизить накладные расходы, связанные с передачей заданий с центрального процессора.

2. Для обработки узлов с достаточно большим числом геометрических примитивов (в данном случае — более 256) используются несколько групп заданий на узел (в данной работе — $N/512$, где N — число геометрических примитивов в узле). В сочетании с оптимизацией из пункта (1) такой подход обеспечивает использование ресурсов, близкое к оптимальному.

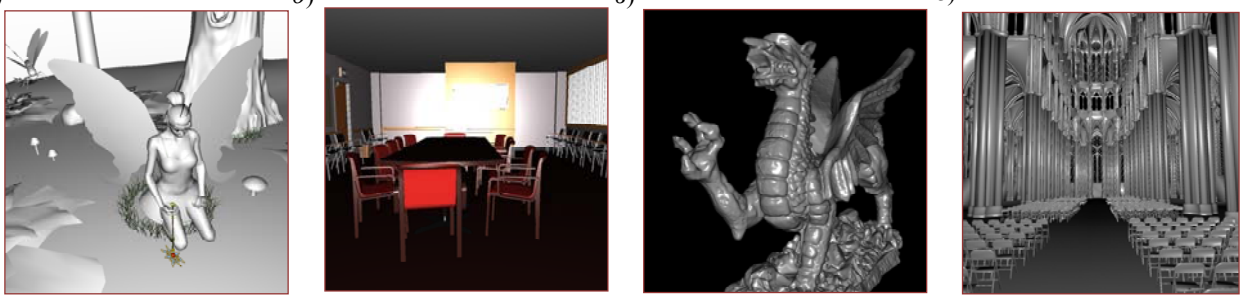
3. Построение дерева включает 3 этапа: обработка больших узлов (более 32K примитивов), обработка средних узлов (от 256 до 32K примитивов) и обработка малых узлов (менее 256 примитивов). Такой подход позволяет использовать специальные модификации алгоритма на каждом уровне, в результате чего удается сократить накладные расходы и оптимизировать использование ресурсов ГПУ.

В целом на центральном процессорном устройстве происходит только отправка существующих заданий и генерация новых, все этапы выполнения алгоритма реализуются на ГПУ.

Экспериментальные исследования производительности алгоритма были проведены на примере тестовых сцен (см. рисунок), характеризуемых различной вычислительной сложностью обработки изображения (содержат от 10K до 3M треугольников). Поскольку основная цель исследований связана с ускоряющими структурами, для задач визуализации в вычислительных экспериментах использовалась классическая трассировка лучей Уиттеда. Для обхода

дерева BVH использовался классический алгоритм на основе полного стека (в частной памяти каждого потока), который позволяет оптимизировать вычисления за счет обработки дерева в порядке от ближних узлов к дальним.

а) *б*) *в*) *г*)



Исследования проводились на компьютере с ГПУ NVIDIA GeForce GTX 480 под управлением операционной системы Ubuntu Linux 11.04 (с версией видеодрайвера 270.41.06). В таблице полученные результаты сравниваются с данными других работ.

Результаты экспериментальных исследований производительности алгоритма построения SAH BVH деревьев на ГПУ

Тестовая сцена		*Lauterbach [3]*		*Wald [4]*		*Разработанный алгоритм*	
Fairy Forest (*а*)	174K	488 мс	21,7 Флоп	31 мс	29 Флоп	40 мс	25 Флоп
Conference (*б*)	284K	477 мс	24,5 Флоп	42 мс	46 Флоп	68 мс	36 Флоп
Welsh-dragon (*в*)	2,2M	—	—	—	—	362 мс	19,6 Флоп
Cathedral (*г*)	3,2M	—	—	—	—	697 мс	14 Флоп

Из таблицы видно, что предложенный алгоритм позволил улучшить результаты более чем в 10 раз, по сравнению с лучшей известной реализацией [3] (с учетом разницы в использованной аппаратуре — в 5 раз). Более того, полученные временные оценки сопоставимы с оценками для „компромиссных“ структур (обеспечивающих быстрое построение, но менее эффективных при трассировке — Hybrid BVH, Two-level Grids), полученными в последних работах [3, 5]. Текущая реализация позволяет выполнять визуализацию произвольных *динамических* сцен до 800K треугольников и *статических* сцен до 5M треугольников.

Работа выполнена в рамках ФЦП „Исследования и разработки по приоритетным направлениям развития научно-технологического комплекса России на 2007—2012 гг.“ и „Научные и научно-педагогические кадры инновационной России на 2009—2013 гг.“.

СПИСОК ЛИТЕРАТУРЫ

1. *Foley T., Sugerman J.* KD-Tree Acceleration Structures for a GPU Raytracer // Proc. of the ACM SIGGRAPH/EUROGRAPHICS Conf. on Graphics Hardware (HWWS'05). Los Angeles, CA, USA, 2005. P. 15—22.
2. *Goldsmith J., Salmon J.* Automatic Creation of Object Hierarchies for Ray Tracing // IEEE Comput. Graph. Appl. 1987. N 7. P. 14—20.
3. *Lauterbach C., Garland M.* Fast BVH Construction on GPUs // Computer Graphics Forum. 2009. N 28. P. 375—384.
4. *Wald I.* Fast Construction of SAH BVHs on the Intel Many Integrated Core (MIC) Architecture [Электронный ресурс]: <http://techresearch.intel.com/spaw2/uploads/files/TR_MIC_SAH_BVHsBuild.pdf>.
5. *Kalojanov J., Billeter M., Slusallek P.* Two-level Grids for Ray Tracing on GPUs // Computer Graphics Forum. 2011 (to be published).

Сведения об авторах

Денис Константинович Боголепов — аспирант; Нижегородский государственный университет им. Н. И. Лобачевского, кафедра математического обеспечения ЭВМ;
E-mail: denisbogol@gmail.com

Дмитрий Петрович Сопин — Нижегородский государственный университет им. Н. И. Лобачевского, кафедра математического обеспечения ЭВМ; ассистент; E-mail: sopindm@gmail.com

Данила Ярославович Ульянов — студент; Нижегородский государственный технический университет им. Р. Е. Алексеева, кафедра вычислительных систем и технологий; E-mail: danila-ulyanov@yandex.ru

Вадим Евгеньевич Турлапов — д-р техн. наук, доцент; Нижегородский государственный университет им. Н. И. Лобачевского, кафедра математического обеспечения ЭВМ; E-mail: vadim.turlapov@gmail.com, vadim.turlapov@cs.vmk.unn.ru

Рекомендована НИИ НКТ

Поступила в редакцию
15.05.11 г.