7. *Zakharov I.V., Terekhov V.G.* Avtonomnoe upravlenie funktsionirovaniem bortovykh vychislitel'nykh sistem na osnove resursnogo podkhoda // Estestvennye i tekhnicheskie nauki. 2017. № 1 (103). S. 113–115.
8. *Ievlev V.I., Filippov G.A.* Kachestvo i nadezhnost' elektronnoj komponentnoj bazy EVM spetsial'nogo naznacheniya. Ekaterinburg: UrFU, 2013. 102 s.
9. *Kirilin A.N. i dr.* Metody obespecheniya zhivuchesti nizkoorbital'nykh avtomaticheskikh KA zondirovaniya Zemli: matematicheskie modeli, komp'yuternye tekhnologii. M.: Mashinostroenie, 2010. 384 s.
10. *Nechaj A.A.* Modelirovanie sistemy upravleniya robototekhnicheskim kompleksom likvidatsii chrezvychajnykh situatsij na osnove mnogomernykh kopula-funktsij // Sovremennye problemy sozdaniya i ekspluatatsii vooruzheniya, voennoj i spetsial'noj tekhniki: sbornik statej III Vserossijskoj nauchno-prakticheskoj konferentsii. SPb., 2016. S. 287–292.
11. *Nechaj A.A.* Formirovanie bezopasnoj informatsionnoj sredy // Aktual'nye problemy sovremennosti: nauka i obshchestvo. 2019. № 4 (25). S. 43–44.
12. *Nechaj A.A., Borisov A.A., Borisova Yu.I.* Tochechnyj analiz dannykh distantsionnogo zondirovaniya Zemli sredstvami yazyka programmirovaniya Python // Vestnik Rossijskogo novogo universiteta. Seriya "Slozhnye sistemy: modeli, analiz i upravlenie". 2019. Vyp. 1. S. 49–55.
13. *Okhtilev M.Yu., Sokolov B.V., Yusupov R.M.* Teoreticheskie i tekhnologicheskie osnovy kontseptsii proaktivnogo monitoringa i upravleniya slozhnymi ob"ektami // Izvestiya YuFU. Seriya: Tekhnicheskie nauki. 2015. Vyp. 1. S. 162–174.
14. *Shajmardanov A.M., Nechaj A.A., Lepekhin S.V.* Matematicheskie modeli sistem avtomaticheskogo upravleniya s shirotno-impul'snoj modulyatsiej // Vestnik Rossijskogo novogo universiteta. Seriya "Slozhnye sistemy: modeli, analiz i upravlenie". 2019. Vyp. 2. S. 27–39.
15. *Ejkkhoff J.* Bortovye komp'yutery, programmnoe obespechenie i poletnye operatsii. M.: Tekhnosfera, 2014. 336 s.

DOI: 10.25586/RNU.V9187.20.02.P.037 УДК 004.925

М.С. Никитенко, Д.А. Ершов, М.В. Раскатова

ОРГАНИЗАЦИЯ ВЫЧИСЛЕНИЙ ДЛЯ ОТСЕЧЕНИЯ ПОВЕРХНОСТЕЙ ПРИ ВИЗУАЛИЗАЦИИ ТРЕХМЕРНЫХ СЦЕН В РЕАЛЬНОМ ВРЕМЕНИ

Рассматриваются особенности организации процесса визуализации трехмерных сцен в реальном времени. Особое внимание уделяется проблеме определения видимости обрабатываемой системой визуализации геометрии. Дается характеристика стандартным методам отсечения и определения видимости, выделяются их недостатки. Описывается современный подход к отсечению невидимых поверхностей и определяются способы его усовершенствования. Рассматриваются различные способы повышения эффективности процесса вычислений при осуществлении отсечения и визуализации за счет иной организации в системах, содержащих несколько гетерогенных графических процессоров. Предлагается использование направленного ациклического графа в качестве высокоуровневого знания о зависимостях между этапами визуализации для эффективного планирования.

Ключевые слова: визуализация в реальном времени, оптимизация вычислений на графических процессорах, множественные графические процессоры.

M.S. Nikitenko, D.A. Ershov, M.V. Raskatova

COMPUTATION PLANNING FOR SURFACES CULLING IN REAL-TIME RENDERING OF 3D SCENES

The features of the organization of the process of rendering of three-dimensional scenes in real-time are considered. Particular attention is paid to the problem of determining the visibility of the geometry processed by the rendering system. Characteristics are given to standard methods of clipping and determining visibility, their disadvantages are highlighted. A modern approach to culling of invisible surfaces is described and ways to improve it are determined. Various ways of increasing the efficiency of the computation process when performing clipping and rendering due to a different organization in systems containing several heterogeneous graphics processors are considered. It is proposed to use a directed acyclic graph as a high-level knowledge of the dependencies between the stages of rendering for effective planning.
Keywords: real-time rendering, GPU program optimization, multi-GPU.

Введение

Визуализация является вычислительной задачей, которая эффективно решается с помощью видеокарт (графических адаптеров). Современные видеокарты содержат в себе собственную оперативную память, подсистему дополнительного питания и прочее, но главное – графический процессор.

Графический процессор – это специализированная интегральная схема, разработанная для эффективной обработки и формирования изображений компьютерной графики. Благодаря многопоточной архитектуре с множеством ядер и специализированной конвейерной архитектуре графические процессоры намного эффективнее в обработке графической информации, чем центральный процессор. Графические процессоры используются не только для расчетов, связанных с построением и обработкой изображений компьютерной графики, но и для различных параллельных вычислений.

Процесс получения изображения по некоторой модели с помощью компьютерной программы называется рендерингом, или визуализацией [1]. В зависимости от цели рендеринга различают предварительную визуализацию и визуализацию в реальном времени. В данной статье внимание будет уделено визуализации в реальном времени.

Визуализация в реальном времени направлена на получение результата за наименьшее время, то есть время формирования очередного кадра визуализации сильно ограничено в силу необходимости осуществления визуализации актуального состояния виртуального пространства с частотой не менее 30 кадров в секунду ввиду особенности восприятия визуальной информации человеческим глазом.

Таким образом, благодаря данной визуализации становится возможным обеспечение интерактивности, создание ощущения взаимодействия с виртуальным миром в реальном времени за счет быстрой реакции системы на вносимые пользователем изменения.

К системам, в которых используется визуализация в реальном времени, относятся: САПР, системы виртуальных и дополненных реальностей (тренажеры, симуляторы, компьютерные игры).

В настоящее время визуализация в реальном времени стремится к достижению фотореализма получаемой картины за счет аппроксимирования физических законов реального мира [1]. Для этого необходимо использовать множество различных этапов визуализации, каждый из которых отвечает за свою составляющую финального изображения. Таким образом, главной задачей визуализации в реальном времени является максимальная оптимизация вычислений, выполняемых при создании одного кадра.

Современные системы визуализации в реальном времени

Системы визуализации стремятся достичь фотореалистичной графики. Для этого необходимо использовать множество различных этапов визуализации, каждый из которых отвечает за свою составляющую финального изображения.

Помимо различных методов, направленных непосредственно на получение фотореалистичной картины, существуют невидимые конечному пользователю этапы, являющиеся неотъемлемой частью процесса визуализации и направленные на его эффективное выполнение.

Например, одной из основных проблем современной визуализации в реальном времени является то, что графический процессор в общем случае не имеет представления о том, что конечный пользователь видит в виртуальном мире на данный момент. В графическом конвейере визуализации присутствует фиксированный этап отсечения невидимой геометрии. На данном этапе выполняется отсечение геометрии, которая вышла за пределы объема отсечения после обработки вершин. Затем дополнительно выполняется отбрасывание растеризованных фрагментов, которые не проходят тесты глубины или маскированы трафаретом.

К сожалению, все эти стадии никак не спасают от бессмысленно потраченных вычислительных ресурсов графического процессора на обработку геометрии в начальных стадиях графического конвейера (вершинный и геометрические шейдеры), которая в конечном итоге оказывается невидима и отбрасывается или отсекается на вышеописанных этапах [1; 4].

Стандартные методы отсечения лишней геометрии до визуализации оказываются неэффективными, так как имеют недостаточную гранулярность.

Самый простой метод отсечения по пирамиде видимости отсекает объекты трехмерной сцены от отправки на визуализацию только в том случае, если ограничивающий объем объекта не пересекается с пирамидой видимости. В современных системах визуализации трехмерные объекты имеют очень большое количество примитивов для высокого качества визуализации. Таким образом, если от огромной трехмерной модели какого-то объекта видна лишь его малая часть, то он все равно будет целиком отправлен на обработку графическим конвейером.

Другие метод отсечения требуют дополнительной информации о трехмерной сцене и визуализируемом кадре, как, к примеру, метод отсечения по пирамиде глубины, которому требуется информация о геометрии с предыдущего кадра визуализации или грубой аппроксимации геометрии текущего кадра.

Однако этого недостаточно, чтобы эффективно избавить графический процессор от лишних вычислений при обработке вершин примитивов, которые не вносят вклад в финальное изображение кадра.

Современный подход к отсечению

Для максимальной эффективности отсечения необходимо взаимодействовать с отдельными примитивами геометрии трехмерной сцены ещё до того, как будет осуществляться непосредственная визуализация этой геометрии или же её использование в определенных методах, которые также требуют определения видимости.

Современные графические процессоры позволяют использовать свои вычислительные ресурсы для осуществления расчетов общего назначения, не всегда связанных напрямую с визуализацией. Более того, подобные вычисления могут выполняться асинхронно с вычислениями, происходящими с использованием графического конвейера, потому как не задействуют его этапы с фиксированной функциональностью. Такие расчеты выполняются с помощью вычислительных шейдеров.

Трехмерные сцены содержат огромное количество трехмерных примитивов (чаще всего треугольников). Вычислительные шейдеры являются наиболее подходящим инструментом для обработки таких объемов данных. За счет асинхронности определение видимости и отсечение можно выполнять с помощью вычислительных шейдеров уже в тот момент, когда визуализируется предыдущий кадр визуализации.

С помощью вычислительных шейдеров возможно выполнение многоуровневого отсечения с различной степенью гранулярности. Благодаря наличию непрямой визуализации возможно создание команд для визуализации непосредственно на графическом процессоре, что позволяет существенно снизить накладные расходы по пересылке данных и перенести большую часть процесса отсечения непосредственно на графический процессор.

Подобный подход к процессу отсечения и определения видимости хорошо ложится на принципы выполнения вычислительных шейдеров на графическом процессоре.

Аналогичный подход применяется в методе «буфер видимых треугольников» [2; 3]. Из названия метода сразу же понятна конечная цель – получить для текущей позиции камеры (будь то наблюдатель или же источники света, от которых учитывается получаемая тень) список видимых треугольников со всеми необходимыми параметрами, чтобы визуализация для каждой точки наблюдения сводилась к эффективному по памяти и по количеству вычислений обходу всех треугольников в порядке размещения в визуализируемом буфере. Однако данный подход направлен в первую очередь на эффективную визуализацию в условиях высокого разрешения финального изображения, тогда как часто наибольшую роль играет не разрешение конечного изображения, а количество и сложность трехмерных объектов виртуальной сцены.

Множественные графические процессоры

Современные графические API (Vulkan и Direct X 12) представляют программистам компьютерной графики больший контроль над взаимодействием и управлением графическим процессором, однако требуют четкого понимания того, как именно будут выполняться и взаимодействовать отдельные этапы визуализации. Помимо этого становится возможным осуществлять управление сразу несколькими графическими процессорами, присутствующими в системе, а также совместно использовать их вычислительные ресурсы. Стоит заметить, что большинство современных систем с дискретными видеокартами содержит также интегрированные графические процессоры в составе центрального процессора.

С использованием нескольких графических процессоров становится возможным осуществить еще более эффективное отсечение за счет распределения вычислительной нагрузки. В такой конфигурации возможно несколько сценариев выполнения процесса визуализации.

Одним из вариантов является выполнение отсечения исключительно на интегрированном графическом процессоре, который будет подготавливать информацию о видимых участках сцены для эффективной визуализации графическим процессором дискретной видеокарты. Данная схема не требует жесткой синхронизации между графическими процессорами и позволяет «разгрузить» основной графический процессор от осуществления отсечения, что, в свою очередь, позволит выполнять более сложные расчеты для получения более качественного изображения.

Другим вариантом взаимодействия является частичный перенос отсечения на интегрированный графический процессор. Однако стоит помнить, что интегрированные графические процессоры чаще всего уступают по возможностям графическим процессорам в составе дискретных видеокарт. В такой ситуации возможно несколько способов для осуществления такого взаимодействия:

- перенос лишь отдельных этапов отсечения на интегрированный графический процессор;
- осуществление отсечения для менее важных положений камеры на интегрированном графическом процессоре (например, для построения карт теней для источников света).

Помимо всего вышесказанного стоит учесть и тот факт, что время, необходимое для выполнения отдельных этапов процессов отсечения и визуализации, зависит от текущей конфигурации трехмерной сцены, которая может изменяться в ходе взаимодействия с пользователем, а также от положения наблюдателя и источников света. При этом в современных системах визуализации нет полноценной обратной связи, которая позволяла бы изменять распределение команд по очередям и по времени выполнения для более эффективного использования ресурсов графических процессоров на основе учета времени выполнения отдельных этапов.

Граф визуализации

Для эффективного использования доступных вычислительных ресурсов необходимо высокоуровневое знание всех зависимостей между этапами визуализации и вычислений, осуществляемых при построении одного кадра визуализации, а также учит текущей загруженности тех или иных этапов для корректного планирования их выполнения.

Этого можно достичь путем представления последовательности этапов визуализации в виде направленного ациклического графа, выполнение которого будет планироваться с учетом множества факторов и распределяться по всем доступным в системе графическим процессорам для максимально эффективного использования доступных вычислительных ресурсов, что потенциально должно привести к уменьшению общего времени визуализации.

Выводы

Необходимо разработать собственный метод определения видимости и отсечения невидимых поверхностей, который сочетал бы в себе преимущества существующих мето-

дов и подходов, был лишен их недостатков, а также предоставлял бы универсальный подход к определению видимости и отсечению, способный эффективно работать в различных типах визуализации.

Помимо этого необходимо формировать и использовать высокоуровневое знание всех этапов визуализации и времени их выполнения. Это позволит осуществлять эффективное планирование выполнения команд и распределения задач между множественными графическими процессорами.

Литература

1. *Akenine-Moller T. et al.* Real-Time Rendering. 4th ed. A K Peters/CRC Press, 2018. 1198 p.
2. *Burns C., Hunt W.* The Visibility Buffer: A Cache-Friendly Approach to Deferred Shading. URL: http://jcgt.org/published/0002/02/04/ (date of the application: 13.03.2020).
3. *Engel W.* Diary of a Graphics Programmer: Triangle Visibility Buffer. URL: http://diaryofagraphicsprogrammer.blogspot.com/2018/03/triangle-visibility-buffer.html (date of the application: 11.03.2020).
4. *Wihlidal G.* Optimizing The Graphics Pipeline With Compute. URL: https://www.ea.com/frostbite/news/optimizing-the-graphics-pipeline-with-compute (date of the application: 10.03.2020).

DOI: 10.25586/RNU.V9187.20.02.P.042 УДК 681.5.017

А.О. Жуков, К.А. Крупский, У.А. Пестун

АДАПТИВНЫЙ АЛГОРИТМ ФИЛЬТРАЦИИ ПАРАМЕТРОВ ДВИЖЕНИЯ КОСМИЧЕСКОГО ОБЪЕКТА НА ГЕОСТАЦИОНАРНОЙ ОРБИТЕ*

Разработан адаптивный алгоритм фильтрации параметров движения космического объекта на геостационарной орбите, сочетающий в себе простоту реализации и высокую точность оценок.
Ключевые слова: алгоритм, динамическая фильтрация, ковариация, фильтр Калмана, моделирование.

A.O. Zhukov, K.A. Krupskij, U.A. Pestun

ADAPTIVE FILTRATION ALGORITHM FOR SPACE OBJECT MOVEMENT PARAMETERS IN A GEOSTATION ORBIT

An adaptive algorithm is developed for filtering the motion parameters of a space object in a geostationary orbit, combining ease of implementation and high accuracy of estimates.
*Keywords: a*lgorithm, dynamic filtering, covariance, Kalman filter, modeling.

* Статья подготовлена при финансовой поддержке гранта Президента России (проект НШ-2686.2020.8 «Модели, методы и средства получения и обработки информации о космических объектах в широком спектральном диапазоне электромагнитных волн»).