ИССЛЕДОВАНИЕ ПРОИЗВОДИТЕЛЬНОСТИ КООРДИНАТНЫХ ПРЕОБРАЗОВАНИЙ ГЕОМЕТРИЧЕСКИХ ОБЪЕКТОВ НА ОСНОВЕ ТЕХНОЛОГИИ ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛЕНИЙ В OPENCL

Мокров А.В.

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Ульяновский государственный технический университет»

Ульяновск, Россия

PERFORMANCE ANALYSIS OF COORDINATE TRANSFORMATIONS OF A GEOMETRIC OBJECT TECHNOLOGY-BASED PARALLEL COMPUTING IN OPENCL

Mokrov A.V.

Ulyanovsk State Technical University

Ulyanovsk, Russia

Проектирование и разработка высокопроизводительных систем, основанных на алгоритмах параллельной обработки координатных данных, является актуальным направлением исследований в области компьютерной геометрии и графики. До недавнего времени решение этой задачи ограничивалось вычислительными возможностями центрального процессора (CPU). Но к настоящему моменту персональные компьютеры оснащены современными видеокартами, которые могут стать решением проблемы обработки трудоемких вычислительных задач, связанных, в том числе, с преобразованиями координатной информации при визуализации большого количества анимационных геометрических объектов на графических процессорах (GPU). В этой связи данная работа, посвященная исследованию эффективности организации параллельных вычислений преобразований координат геометрических объектов в задачах визуализации трехмерных сцен на CPU и GPU средствами OpenCL, является актуальной.

Целью данной работы является исследование и оценка использования графических процессоров для решения задач преобразований координат геометрических объектов в процессе визуализации трехмерных сцен на основе анализа технологии организации параллельных

вычислений ресурсоемких процессов, а так же составление алгоритмов, позволяющих увеличить быстродействие вычислений координатных преобразований для GPU.

Композиция аффинных преобразований (поворот вокруг осей, параллельный перенос, масштабирование) при визуализации трехмерных объектов сцены реализуется, как правило, через вычисления произведения соответствующих результирующих матриц преобразования на вектор {x, y, z}, определяющий координаты соответствующей точки. Это, как правило, существенно замедляет процесс визуализации вследствие большей сложности алгоритма [2, 5].

Применительно к программированию графического конвейера API OpenGL версии 4.0 и выше, из стандарта исключены функции по осуществлению матричных вычислений, и возложены на программируемые шейдеры. В этой связи традиционный подход, заключающийся в определении новых координат множества мочек через последовательное умножение матриц элементарных аффинных преобразований, в настоящий момент, с выходом новой спецификации OpenGL 4.0, не может быть реализован непосредственно.

Отметим, что в ранних версиях OpenGL обеспечивалась поддержка преобразований координат и проекций на основе использования стандартного стэка матриц (GL_MODELVIEW и GL_PROJECTION). В ядре OpenGL 4.0 функциональность, поддерживающая матричный стэк, была удалена. Следовательно, в задачах визуализации геометрических преобразований возникает потребность обеспечивать поддержку матриц преобразования и проекций, а затем передавать их в шейдеры графического конвейера.

Для решения этой задачи на современном этапе можно использовать библиотеку GLM (OpenGL Mathematics) [4] на языке C++ для выполнения математических операций в 3D-программах, базирующуюся на спецификации OpenGL Shading Language (GLSL). Для сравнения производительности вычислений композиции преобразований будем пользоваться версией библиотеки GLM 0.9.4.0 [5] с поддержкой Visual Studio 2010.

Альтернативным подходом полному векторно-матричному умножению является прямое вычисление координат [2]. При этом координаты вершин множества точек {x, y, z} претерпевают композицию преобразований, выраженных соответствующими матрицами масштабирования, переноса, поворота и рассчитываются независимо.

Проанализируем возможные пути параллельного вычисления преобразований в разработанных модулях на CPU или GPU с использованием единого стандарта описания вычислений на высокопараллельных системах OpenCL (Open Computational Library).

Следует отметить, что при разработке параллельных алгоритмов решения вычислительных задач принципиальным моментом является анализ эффективности

использования параллелизма, состоящий обычно в оценке получаемого ускорения процесса вычисления (сокращения времени решения задачи). Формирование подобных оценок ускорения может осуществляться применительно к выбранному вычислительному алгоритму (оценка эффективности распараллеливания конкретного алгоритма). Другой важный подход может состоять в построении оценок максимально возможного ускорения процесса получения решения задачи конкретного типа (оценка эффективности параллельного способа решения задачи). В настоящем исследовании в качестве критерия эффективности решения поставленной задачи будем использовать второй подход.

В качестве вычислительной системы для экспериментального сравнения производительности рассмотрим компьютер, построенный на базе процессора со следующими характеристиками:

1) номер процессора – i7-2670QM;

2) количество ядер – 4;

3) количество потоков – 8;

4) тактовая частота – 2.2 GHz;

Основные технические характеристики видеокарты GeForce GT 540M:

1) графический процессор – GF108;

2) количество универсальных процессоров – 96;

3) количество блоков – 16 текстурных и 14 блоков блендинга;

4) тактовая частота ядра – 672 МГц;

5) тактовая частота универсальных процессоров – 1344 МГц;

8) объём памяти – 1024 Мбайта;

9) шина памяти – 128 бит.

10) поддержка DirectX 11, включая шейдеры версии 5.0, OpenGL 4.1, DirectCompute 11 и OpenCL 1.0.

Рассмотрим реализацию двух алгоритмов для вычисления матричных преобразований в OpenCL: программу непосредственного вычисления координат после преобразования, и, также, с использованием средств вычисления матриц библиотеки GLM версии 0.9.4.0. Результаты тестирования производительности представлены на рис. 1-2.

Так, при вычислении координат 1024x1024 точек параллельно на четырех ядрах CPU время расчета более, чем в 30 раз меньше времени аналогичного расчета на хосте. Параллельная реализация процесса на GPU позволяет практически в два раза ускорить этот процесс.

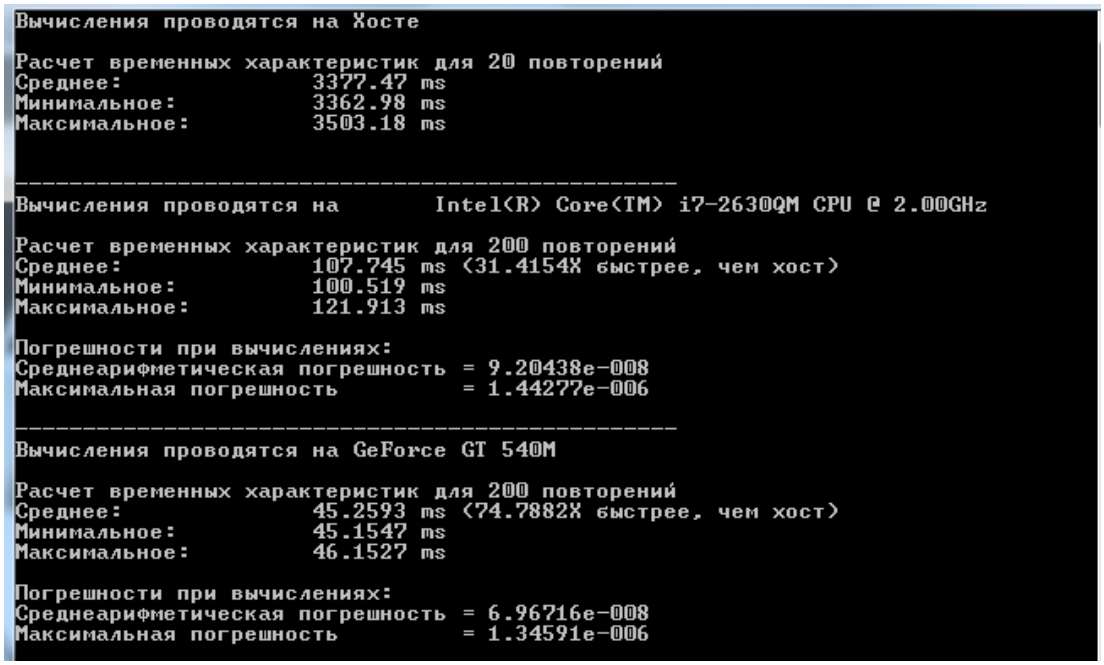

Рис. 1. Результаты теста параллельного вычисления координатных преобразований в OpenCL

Однако при дальнейшем росте количества точек быстродействие программы изменяется практически линейно. В некоторых случаях оптимизация быстродействия программы оказалась существенной. Однако оптимизация осуществлялась без учета особенностей аппаратной реализации вычислений. Вместе с тем, существенное повышение производительности вычислительного процесса возможно за счет использования специальной модели памяти, принятой в OpenCL [3].

Согласно этой модели, как только данные переносятся с хоста на устройство (device), они остаются в глобальной памяти. Любые данные, переносимые в обратном направлении, также "оседают" в глобальной памяти хоста. Адресное пространство локальной памяти уникально для каждого устройства. Доступ к этой памяти имеет гораздо меньшую латентность и, следовательно, она обладает гораздо более высокой пропускной способностью, чем глобальная память. Тестирование оптимизированного кернела на CPU и GPU заметного прироста в быстродействии не показало. Это с одной стороны, обусловлено реализацией обработки координат вершин в виде линейных массивов, а с другой стороны, достаточной величиной шины памяти GPU (128 бит).

Вместе с тем, существенный прирост производительности возможен при вычислении более сложного алгоритма умножения квадратных матриц, реализованного с помощью библиотеки OpenGL Mathematics. Тестирование подтвердило, что указанный алгоритм требует значительно больших вычислительных затрат при непараллельном вычислении, более чем в 20 раз превосходящих вычисления по алгоритму прямого вычисления преобразованных координат (рис. 2).

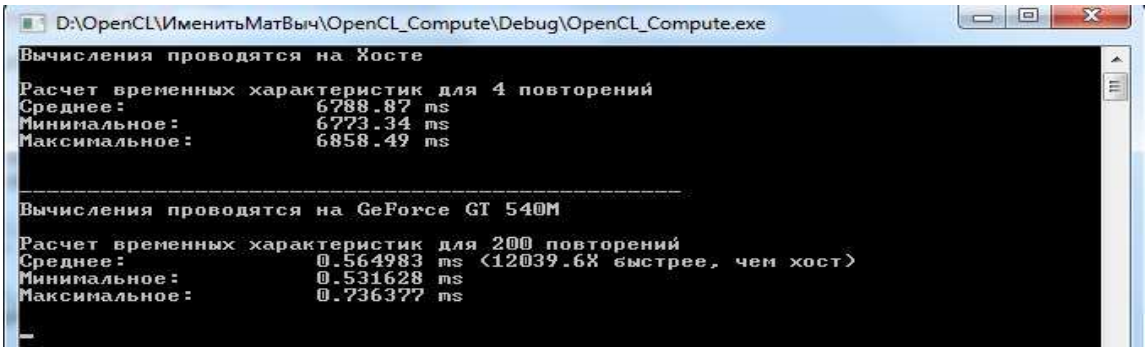

Рис. 2. Результаты теста параллельного вычисления матричных преобразований средствами OpenGL Mathematics в OpenCL

Вместе с тем, параллельное вычисление матриц на GPU позволило за счет оптимизации организации памяти в кернеле достичь существенного скачка производительности процесса.

Таким образом в настоящей работе установлено, что перенос вычислений с центрального процессора на графический может существенно увеличить скорость выполнения ресурсоемких вычислительных задач преобразования координатной информации визуализируемых объектов в трехмерном геометрическом моделировании.

Однако, не всякая реализация алгоритма на графическом процессоре дает прирост в скорости по сравнению с реализацией на центральном процессоре. Параллельные вычисления координатных преобразований без использования операции умножения матриц преобразований на GPU возможны и реализуемы. При этом увеличение объемов пересылаемых данных (количество вершин) практически не влияет на эффективность процесса в сравнении с вычислениями на CPU, вследствие использования линейных массивов для организации хранения координатной информации в глобальной памяти OpenCL.

Реализация умножения матриц средствами библиотеки OpenGL Mathematics, наоборот, существенно замедляет работу приложения. Поэтому при организации сложных вычислений

средствами OpenGL Mathematics целесообразно при переносе алгоритма на графический процессор организовывать хранение координатных данных в локальной памяти устройства.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Нечай, О. Мобильные графические процессоры NVIDIA [Электронный ресурс]. – Компьютерра. 2011. – Режим доступа: http://www.computerra.ru/terralab/platform/622259/.

2. Лихачев, В.Н. Создание графических моделей с помощью Open Graphics Library. – ИНТУИТ, 2011. – Режим доступа: http://www.intuit.ru/department/graphics/grogl/5/3.html.

3. Запрягаев, С. А. Применение графического процессора в ресурсоемких вычислениях на базе библиотеки OPENCL / С.А. Запрягаев, А.А. Карпушин. – Вестник ВГУ, СЕРИЯ: Системный анализ и информационные технологии. 2010, –№ 2. – С.79-87.

4. OpenGL Mathematics (GLM)/ OpenGL SDK: Documentation, Sample Code, Libraries, and Tools for creating OpenGL-based Applications. -2012. - – Режим доступа: http://www.opengl.org/sdk/

5. OpenGL Mathematics 0.9.4.0. – 2012. – Режим доступа:http://www.opengl.org/sdk/.

6. Конструирование и оптимизация параллельных программ / Под ред. профессора, чл.-корр. РАЕН В.Н. Касьянова. – Новосибирск, 2008. – 334 с.