УДК 519.688

ШЕЙДЕРНАЯ ОБРАБОТКА ИЗОБРАЖЕНИЙ С ИСПОЛЬЗОВАНИЕМ БИБЛИОТЕКИ OPENGL И ЯЗЫКА GLSL

Полев А.В.

Научный руководитель – профессор Царев С.П.

Сибирский федеральный университет

OpenGL (open graphics library – открытая графическая библиотека) – набор спецификаций, определяющих платформо- и аппаратно-независимый интерфейс для вывода 3D и 2D графики. По сути, состоит из большого набора функций для управления графическим адаптером, как конечным автоматом. Все команды OpenGL или добавляют какие-либо данные на конвейер визуализации или меняет состояние отдельных частей системы.

Спецификация OpenGL пересматривается Консорциумом ARB (Architecture Review Board), который был сформирован в 1992 году. Консорциум состоит из компаний, заинтересованных в создании широко распространённого и доступного API.

Из-за большого количества компаний в консорциуме OpenGL развивался медленно, в отличие, например, от DirectX на Windows. Однако версия 2.0 задала новые пути развития OpenGL, официально введя поддержку высокоуровневых шейдеров на языке GLSL. Отметим, что OpenGL позволяет активно развиваться, текущая версия спецификации – 4.1.

Шейдер – маленькая программа, исполняемая на видеокарте и работающая с определенными типами данных. Сразу отметим, что видеокарта – узкоспециализированное устройство, поэтому она намного быстрее оперирует числами с плавающей запятой, нежели чем центральный процессор. Поэтому предпочтительнее как можно больше расчетов переносить на нее.

Целью любого графического ПО является определение итогового цвета каждого пикселя. Раньше для этого использовался фиксированный набор операций, что ограничивало программиста. Теперь же, когда некоторые части конвейера визуализации сделали программируемыми, возможности для визуализации расширились.

Сейчас в языке GLSL существуют следующие типы шейдеров:

- Вершинные
- Фрагментные
- Геометрические
- Контроля тесселяции

Последние 2 типа появились сравнительно недавно, не будут рассмотрены в рамках данного доклада, т.к. не используются в демонстрационной программе.

Вершинные шейдеры, как нетрудно догадаться из названия, получают на вход вершины примитива, их главной задачей является определения координат вершины в экранных координатах.

Фрагментные шейдеры будут рассматриваться всю оставшуюся часть доклада. Они получают на вход интерполированные данные из вершинного шейдера, и выполняются для каждого фрагмента изображения (который, в случае успешного прохождения ряда тестов, станет пикселем)

Перейдем к конкретным примерам, и посмотрим, что можно, а что нельзя делать при помощи шейдеров, и какая выгода от их использования.

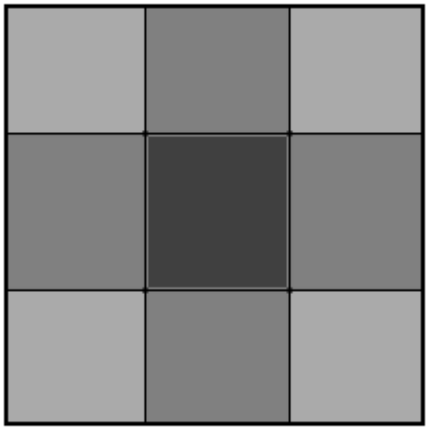
Первый пример – весьма прост, обесцветим изображение:

Любой человек может сказать, что все это можно делать в любом графическом редакторе, но преимущество шейдеров в том, что они используют аппаратную обработку, поэтому различные эффекты мы можем накладывать «на лету». То, что показывается в данном докладе, называется пост-обработкой. Т.е. имея любые графические данные, мы можем применить разные эффекты. Такая обработка применяется для наложения эффектов в кинематографе, для создания реалистичной картинки, если мы моделируем трехмерные миры, для акцентирования внимания зрителя на определенных частях изображения и пр.

Рассмотрим целую группу эффектов, которые базируются на одном принципе. Имеется матрица, называемая ядром свертки (convolution kernel), или же просто – линейным фильтром.

Далее производится выборка n пикселей из изображения (один текущий обрабатываемый и n-1 соседних), значение цвета соответствующего выбираемого

пикселя умножается на соответствующий элемент ядра, после чего все n значений суммируются.



$$\begin{pmatrix} k1 & k2 & k3 \\ k4 & k5 & k6 \\ k7 & k8 & k9 \end{pmatrix}$$

Здесь цветом показан вклад каждого пикселя, например, в фильтре Гаусса (чем темнее, тем значительнее вклад)

Меняя ядро, можно получить ряд различных эффектов – от поиска границ до барельефа.

Рассмотрим еще один интересный эффект – цветокоррекцию, которая повсеместно используется в редактировании видео. Позволяет подчеркнуть нужную атмосферу, изменить контрастность изображения. Отлично ложиться на шейдеры.

Однако шейдер не обязательно должен выполняться в один проход, существуют многопроходные шейдеры. Алгоритм работы следующего примера довольно прост – в первом проходе мы выбираем яркие фрагменты, записывая их в отдельную текстуру (здесь уже используются довольно современные возможности видеоадаптера). Далее, во втором проходе, мы смешиваем их с

исходным изображением, умножая перед этим на константу (чтобы не получить переполнение канала и чрезмерное засветление)

В следующем примере демонстрируется попытка нейтрализовать размытие в изображении. Алгоритм базируется на уже рассмотренных выше примерах поиска границ и повышения резкости. Следующее изображение демонстрирует его работу. Слева направо: исходное изображение - размытое - восстановленное.

В заключение скажем, что шейдеры дают все возможности для решения задач, возникающих в компьютерной графике, позволяя в реальном времени обрабатывать граф