10.7256/2305-6061.2013.01.10

А.Ю. Сморкалов

МАТЕМАТИЧЕСКАЯ И ПРОГРАММНАЯ МОДЕЛИ ГЕНЕРАЦИИ ТЕКСТУР НА ГРАФИЧЕСКИХ ПОТОКОВЫХ ПРОЦЕССОРАХ

Аннотация. *В статье предлагаются обобщенные математическая и программная модели генерации статических и динамических текстур на потоковых процессорах. Математическая модель позволяет оценить эффективность различных подходов к генерации текстур. Предложенная методика обеспечивает высокую производительность за счет возможности выбора оптимального метода генерации и позволяет генерировать не только предопределенные разработчиком динамические текстуры, но и произвольные динамические текстуры, данные для воспроизведения которых поступают из внешнего источника.*

Ключевые слова: *Программное обеспечение, потоковые процессоры, обработка изображений, постэффекты, графические процессоры, вейвлет-преобразование, генерация текстур, статические текстуры, динамические текстуры, процедурные материалы*

1. Введение

Текстура представляет собой растровое изображение, используемое в 3D-графике для наложения на 3D-модели. При хранении в видеопамяти текстура занимает от 32 до 4 бит на 1 пиксель, в зависимости от числа каналов и использования формата без потери или с потерей качества. При хранении в файловой системе с использованием форматов изображения с потерей качества (например, JPEG2000) можно добиться результата в 1-2 бита на пиксель при визуально неизменившемся качестве изображения, однако при решении ряда задач занимаемый текстурами объем данных все равно оказывается чрезмерно большим.

Генерация текстур позволяет сократить объем данных, необходимых для воспроизведения изображения, в сотни и даже тысячи раз. Такая технология имеет большие перспективы применения в целом ряде 3D-приложений, например, для обучающих мультимедиа-программ [1] и виртуальных образовательных миров, т.к. сокращает объем данных необходимых для загрузки через сетевой канал. Кроме того значительный потенциал применения имеет генерация текстур при необходимости в многопользовательской среде синхронизировать не только отдельные свойства объектов [2], но и полностью одно или несколько меняющихся изображений,

10.7256/2305-6061.2013.01.10

передаваемых одним из пользователей. Однако существенным недостатком этого подхода является значительное время, требуемое на преобразование данных для генерации текстуры в растровое изображение.

Графические потоковые процессоры (далее ГПП) обладают большим потенциалом для решения задачи генерации текстур, обладая в сотни раз большим быстродействием, чем центральный процессор (далее ЦП), на котором эти задачи решались ранее. Поэтому актуальной является разработка математической и программной модели (архитектуры) для генерации текстур на ГПП.

2. Генерация текстур и её виды

В общем случае различают генерацию статических и динамических текстур. Под статической текстурой понимают изображение, не меняющееся с течением времени. Генерация статических текстур на ГПП исследована для частных случаев, например, в [3], однако обобщенной модели генерации статических текстур не представлено. Под обобщенной моделью будет понимать модель, объединяющую в себе несколько подходов к генерации текстур.

При генерации статических текстур возможны три подхода. Первый предполагает генерацию текстуры на основе некоторого начального изображения и одного или нескольких примененных к нему фильтров. Второй предполагает растеризацию векторных фигур для генерации текстуры. Третий, комбинированный метод, сочетает в себе первый и второй подходы.

Под динамической текстурой понимают изображение, меняющееся с течением времени. Генерация динамических текстур на графических потоковых процессорах была реализована для различных частных случаев, наиболее известными из которых являются процедурные материалы [4] и постэффекты [5]. Однако, следует отметить, что обобщенной модели генерации динамических текстур на ГПП пока не представлено. Также не исследовались возможности представления генерации динамической текстуры как серии статических и условия целесообразности такого представления.

В случае использования процедурных материалов текстура не генерируется полностью, вместо этого при рендеринге каждого кадра 3D-сцены рассчитывается только цвет пикселей текстуры, необходимых для отображения 3D-объекта при текущем положении виртуальной камеры и текущей матрицы моделирования вида объекта. В некоторых случаях имеет смысл повторяемая с определенной периодичностью генерация статической текстуры и использование её при отображении 3D-объекта. Такой подход существенно сокращает вычислительные затраты по сравнению с использованием процедурных материалов.

Например, на рис. 1. представлен пример отображения динамического неба на основе 3D-сферы с наложенной на нее сгенерированной текстурой. В зависимости от частоты изме-

10.7256/2305-6061.2013.01.10

нения текстуры, необходимой для отображения неба, наиболее оптимальным является первый или второй из описанных подходов.

Рис. 1.
Генерация текстуры динамического неба на ГТП в проекте vAcademia [6]

Особый интерес представляют собой динамические текстуры, данные для воспроизведения которых поступают из внешнего источника по сетевому каналу. Их содержимое никоим образом не предопределено разработчиком приложения. Примером применения может служить инструмент “screen sharing” в образовательном виртуальном мире vAcademia, позволяющий учителю продемонстрировать студентам в динамике часть своего экрана с какой-либо программой или видео с веб-камеры (рис. 1, правый экран).

Для описанных динамических текстур имеет большой потенциал использование прямого и обратного дискретных вейвлет-преобразований (далее ДВП), позволяющих преобразовать изображение в набор данных существенно меньшего размера и восстановить по ним изображение с умеренной потерей качества. Стоит отметить, что скорость работы ДВП на центральном процессоре зачастую недостаточна для интерактивного применения, особенно в составе ресурсоемких приложений, активно использующих ЦП. Исходя из этого, актуальным является реализация ДВП на ГПП.

В работе [2] была предложена реализация ДВП на основе шейдера, а в работе [3] было проведено сравнение применения схемы каскада банков и схемы лифтинга для реализации ДВП на ГПП. В перечисленных работах было достигнуто умеренное ускорение (до 6 раз) по сравнению с реализацией на ЦП, а сама реализация требовала обширных знаний, специфичных для ГПП, т.к. не опиралась на какую-либо обобщенную математическую модель и программную архитектуру.

10.7256/2305-6061.2013.01.10

Таким образом, вычислительные возможности ГПП позволяют осуществлять генерацию текстур в десятки раз быстрее, однако обобщенной модели генерации текстур на потоковых процессорах не представлено. В статье предлагается обобщенная математическая модель и программная архитектура обработки изображений на ГПП, позволяющая:

- организовать генерацию статических и динамических текстур, в том числе реализацию процедурных материалов и постэффектов, генерацию динамических текстур с непредопределенным содержимым.
- сравнить и оценить оптимальность подходов к генерации текстур.
- реализовать все три вышеописанных подхода к генерации статических текстур.

3. Математическая модель

Будем рассматривать изображение в цветовой модели RGBA:

$$U(x, y) = \{f_R(x, y), f_G(x, y), f_B(x, y), f_A(x, y)\}, \quad (1)$$

где $f_R(x, y)$, $f_G(x, y)$, $f_B(x, y)$, $f_A(x, y)$ — это дискретные функции, заданные табличным методом. $f_R(x, y)$ представляет красный канал изображения, $f_G(x, y)$ – зеленый, $f_B(x, y)$ – синий, $f_A(x, y)$ – альфа-канал. Значения этих функций лежат в диапазоне [0, 1].

Результатом преобразования G изображения A на основе изображения B будем называть R = G(A, B, x, y) (2), где A — это исходное изображение, B – растеризуемая фигура, G — преобразующая функция. Более подробно данная математическая модель обработки изображений описана в [5].

Такая модель позволяет определить цвет каждой точки сгенерированной текстуры в отдельности, т.к. результирующее изображение является функцией от координат точки. Модель подходит как для задачи генерации полного изображения текстуры, так и для реализации процедурных материалов. Обобщенность модели позволяет сравнить эффективность двух подходов к генерации динамических текстур.

При генерации полного изображения на ГПП время расчета текстуры определяется по формуле

$$T_{РТ} = T_{ПР} + T_1 * W * H \quad (3), \text{ где}$$

$T_{РТ}$ – время генерации полного изображения,

$T_{ПР}$ – время подготовки к преобразованию,

T_1 – время расчета цвета одного пикселя, т.е. расчета (2),

W, H – высота и ширина рассчитываемого изображения.

При расчете цвета пикселей для метода процедурных материалов время расчета определяется как

$$T_{ПМ} = T_1 * K_{ВП} * K_К \quad (4), \text{ где}$$

$T_{ПМ}$ – время на генерацию цвета точек процедурного материала,

$K_{ВП}$ – количество точек, необходимых для визуализации 3D-объекта с процедурных материалом,

10.7256/2305-6061.2013.01.10

$К_К$ – максимальное количество кадров, которое содержимое динамической текстуры может не меняться.

Отношение $Т_{РТ} / Т_{ПМ}$ дает возможность оценить эффективность использования того или иного подхода. Стоит заметить, что по мере увеличения (W * H) отношение $Т_{РТ} / Т_{ПМ}$ стремится к (W * H) / ($К_{ВП}$ * $К_К$).

Описанная математическая модель напрямую подходит для вышеописанных двух первых методов генерации статических текстур. Расширим её для возможности применения комбинированного подхода генерации статических текстур. Согласно используемой модели растеризация представляет собой преобразование, результатом которого является изображение:

$U(x, y) = G_R(G_P(S, M_P))$ (5), где

G_R – растеризующее преобразование, M_P – матрица проецирования, G_P — проецирующее преобразование.

В случае комбинированного подхода растеризация должна происходить на уже сгенерированное растровое изображение, что соответствует

$R_Q = G_Q(A, B, x, y) = G_Q(A, B, x, y) * B_A + A * (1 – B_A)$ (6), где

Q – канал изображения, A – изображение, сгенерированное через последовательность фильтров, B – изображение, сгенерированное путем растеризации векторных фигур.

Подставляя в формулу (6) выражение (5), получаем преобразование, соответствующее комбинированному подходу генерации статических текстур

$R_Q = G_Q(A, U, x, y) = G_Q(A, U, x, y) * U(x, y)_A + A * (1 – U(x, y)_A)$ (7)

Предложенная математическая модель позволяет реализовать несколько подходов к генерации статических и динамических текстур, оценить их эффективность и выбрать среди них оптимальный. На основе этой модели была построена программная архитектура генерации текстур.

4. Программная архитектура

Программная архитектура обработки изображений на ГПП на основе математической модели, анализа архитектуры и особенностей потоковых процессоров и возможностей OpenGL подробно описана в [5]. В основе модели лежат четыре основных объекта (Texture, Drawing Target, Filter и Filter Sequence) и ограничение <β>: изображение-результат и изображения-параметры одного и того же фильтра не могут совпадать. Это условие заложено в модель в целях обеспечения корректности выполнения преобразований на потоковых процессорах, т.е. для организации независимой параллельной обработки фрагментов изображения.

Texture – это изображение в форме (1), хранимое в памяти графического потокового процессора. Drawing Target – это объект, который определяет изображение-результат и цветовую маску. Filter – задает преобразование изображения (2) и в общем случае представляет собой функцию с набором предопределенных и пользовательских параметров,

10.7256/2305-6061.2013.01.10

возвращающую цвет точки для заданных координат. Filter Sequence – задает последовательность нескольких фильтров с их параметрами, в общем случае позволяя организовать конвейер обработки.

Рассмотрим, как используются объекты программной архитектуры для генерации текстур. Генерируемая текстура в этой модели соответствует объекту Texture. При полной генерации текстуры отрисовка финального изображения ведется с настройками объекта Drawing Target, применение которых соответствует времени $T_{ПР}$ в математической модели. При использовании метода процедурных материалов Drawing Target не используется, а объект-результат Texture становится логическим, т.е. не представленным на аппаратном уровне. Объекты типа Texture, необходимые для генерации результирующей текстуры, остаются представленными на аппаратном уровне в любом случае.

Преобразование, генерирующее текстуру, соответствует Filter при использовании метода процедурных материалов или Filter/Filter Sequence при использовании остальных методов. Предопределенная функция GetColor соответствует преобразованию (2) и обеспечивает обобщенность программной архитектуры, позволяя использовать одну и ту же модель, как для метода процедурных материалов, т.к. и для метода периодической полной регенерации текстуры.

Таким образом, предложенная программная архитектура подходит для реализации всех рассматриваемых в статье подходов к генерации текстур, кроме комбинированного метода генерации статических текстур. Для поддержки комбинированного метода генерации статических текстур программная архитектура была расширена объектом LayeredPicture (см. рис. 2).

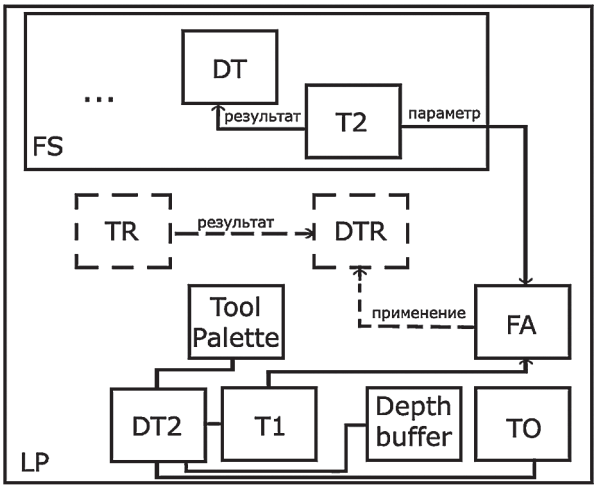

Рис. 2.
Объект LayeredPicture. Пунктирными линиями обозначены опциональные подобъекты.

Объект LayeredPicture состоит из подобъектов FS типа FilterSequence, ToolPalette с набором связанных с ним пообъектов типа Texture (T1 – текстура-результат и TO – временная текстура), DepthBuffer и DrawingTarget (DT2), а также предопределенного объекта FA типа Filter (фильтр альфа-смешивания, соответствующий выражению (6)). В FS на

10.7256/2305-6061.2013.01.10

рис.2. выделены подобъекты DT типа DrawingTarget и T2 типа Texture. T2 хранит в себе результирующее изображение FS.

Опционально в LayeredPicture входит подобъект TR типа Texture и DTR типа DrawingTarget. В случае генерации статической текстуры, TR хранит в себе результирующее изображение. В случае использования метода «процедурных материалов», подобъекты TR и DTR физически отсутствуют, оставаясь логическими.

Таким образом, предложенная обобщенная программная архитектура позволяет осуществлять генерацию статических и динамических текстур, в том числе реализовать процедурные материалы, постэффекты и все три вышеописанных подхода к генерации статических текстур.

5. Реализация вейвлет-преобразований

На основе предложенной в статье математической модели и программной архитектуры можно построить реализацию 1D и 2D ДВП по схеме каскада фильтров. В этом случае прямое вейвлет-преобразование можно разделить на 2 этапа.

Первый этап выполняется объединенным фильтром. Вначале осуществляется преобразование из цветового пространства RGB в YUV. Затем часть изображения, содержащая каналы UV, уменьшается в N раз с осреднением цвета (т.к. человеческий глаз слабо различает цветовые составляющие по сравнению с яркостной). После этого данные каналов UV проходят квантование с коэффициентом D по формулам (8).

$$U = U / D,\ V = V / D \qquad (8)$$

На втором этапе для каждого цветового канала отдельно применяется непосредственно ДВП по схеме каскада фильтров, K – количество проходов. Критерием отброса пикселя с координатами (x,y) как незначительного на I-ом проходе каскада фильтров было удовлетворение той из систем неравенств (9) и (10) в 2D-варианте преобразования, в которой I примет минимальное значение или системе (9) в случае равенства I в обоих системах неравенств. При осуществлении описанной проверки конфликта с другими шагами каскада фильтров не происходит, т.к. в силу <β> изображение-результат и изображение-параметр различаются.

$$\begin{cases} \left| C(x,y) - (C(x - 2^{I+1}, y) + C(x + 2^{I+1}, y)) \right| < E \\ x \bmod 2^{I+1} = 2^{I} \end{cases} \qquad (9),$$

$$\begin{cases} \left| C(x,y) - (C(x, y - 2^{I+1}) + C(x, y + 2^{I+1})) \right| < E \\ y \bmod 2^{I+1} = 2^{I} \end{cases} \qquad (10),$$

где C(x,y) – значение обрабатываемого цветового канала в точке (x,y), а E — допустимое отклонение апроксимированного значения цветового канала пикселя от точного. Если значение цветового канала пикселя отбрасывается, то оно обнуляется (в изображение-результат записывается 0).

10.7256/2305-6061.2013.01.10

Шаг каскада фильтров I может быть найден прямым перебором. Такой подход оправдан, так как позволяет объединить шаги каскада фильтров в один этап, а стоимость избыточных арифметический операций значительно меньше стоимости дополнительных проходов, каждый из которых занял бы время $T_{ПР}$.

Обратное вейвлет-преобразование также должно состоять из (2*K + 1) и (K + 1) этапов для 2D и 1D случаев соответственно. В 1D-случае первые K этапов представляют собой последовательность фильтров, реализующую K шагов каскада фильтров обратного ДВП. В связи с необходимостью использовать на более поздних шагах результаты более ранних и ограничением <β>, объединить обратное ДВП в 1 проход не удалось. Вместо этого используется filter sequence типа «пинг-понг» (см. рис. 3), при котором для всех K этапов используется один и тот же фильтр с разным коэффициентом шага каскада фильтров и чередующимися изображением-результатом и исходным изображением.

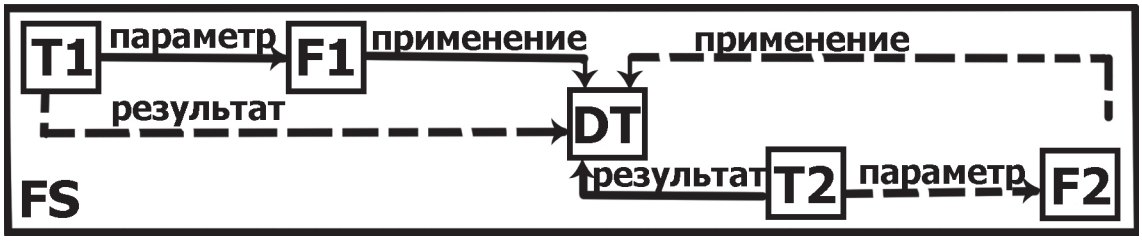

Рис.3. Взаимоотношения объектов в системе на пример filter sequence FS типа «пинг-понг». Сплошные линии соответствуют первому фильтру последовательности, пунктирные второму.

Цвет пикселя с координатами (x,y) для I-го этапа, если существует целое I, удовлетворяющее условию

$$x \bmod 2^{I+1} = 2^{I} \quad (11)$$

определяется выражением

$$C(x,y) = \begin{cases} C(x, y), \text{если } C(x, y) > 0 \\ (C(x - 2^{I+1}, y) + C(x + 2^{I+1}, y)/2, \textit{если } C(x,y) = 0 \end{cases} \quad (12),$$

На последнем этапе происходит деквантизация каналов UV, восстановление исходных размеров изображения по этим каналам и преобразование цвета из пространства YUV в RGB.

6. Результаты

Описанные модели были апробированы в виртуальном образовательном мире vAcademia на ряде задач генерации текстур. Как исходные данные для генерации, так

10.7256/2305-6061.2013.01.10

и результирующая текстура хранились в памяти ГПП. В этом случае достигается наибольший эффект при генерации текстур на потоковых процессорах, т.к. исключается этап копирования данных по шине PCI-E.

Обобщенная модель генерации текстур позволяет напрямую сравнить метод процедурных материалов и метод периодической регенерации текстуры как статической. Сравнение на примере генерации текстуры динамического неба (рис. 1) проводилось с использованием ГПП NVidia GF8800GT (табл. 1). T – время, между двумя последовательными обновлениями динамической текстуры. Сравнение показывает, что эффективность метода процедурных материалов возрастает с уменьшением времени обязательного обновления содержимого динамической текстуры, что соответствует приведенной математической модели.

Табл. 1. Сравнение методов генерации динамической текстуры

T, мс	50	200	500	1000	2500	5000
$T_{ПМ}$ / $T_{РТ}$	0.16	0.71	1.63	3.12	7.31	14.53

Генерация статических текстур методом применения серии фильтров была апробирована на примере генерации уникальных текстур кожи аватаров. Преимущество подхода на ГПП иллюстрирует табл. 2. Сравнение проводилось с использованием ГПП NVidia GF8800GT (соответствует $T_{ГПП}$) и ЦП Intel Core 2 Quad 2.4 ГГц (1 ядро, соответствует $T_{ЦП}$).

Табл. 2. Сравнение времени генерации текстуры кожи аватара

Размер	128x128	256x256	512x512	1024x1024	2048x2048
$T_{ГПП}$, мкс	94	151	280	1041	2001
$T_{ЦП}$, мкс	495	1654	6032	20853	79504

На графиках (рис. 4 и 5) приведено сравнение времени прямого и обратного 2D ДВП на ЦП и ГПП. В качестве ГПП использовался NVidia GF8600 GT. В качестве ЦП выступал Core 2 Duo 2.2 ГГц. S – размер изображения в пикселях.

Из анализа приведенных данных видно, что преимущество ГПП при прямом ДВП достигает ~70 раз, а при обратной ~28 раз. Максимальное преимущество достигается при больших размерах изображения. Более низкая скорость работы ГПП при обратном ДВП объясняется большим числом проходов. Существенной величиной времени $T_{ПР}$ объясняется и нелинейная зависимость времени обработки на ГПП от размера изображения.

10.7256/2305-6061.2013.01.10

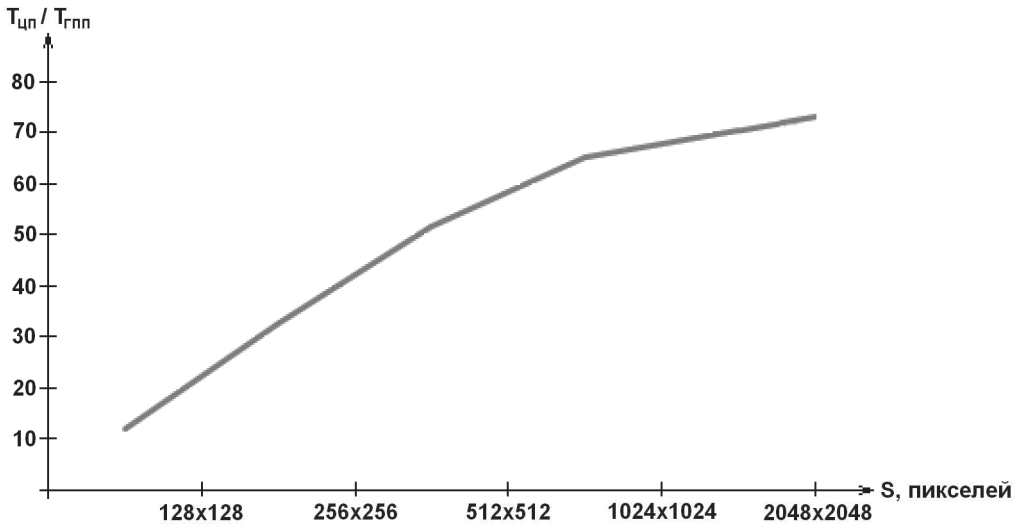

Рис. 4. Отношение времени выполнения прямого 2D ДВП на ЦП и ГПП.

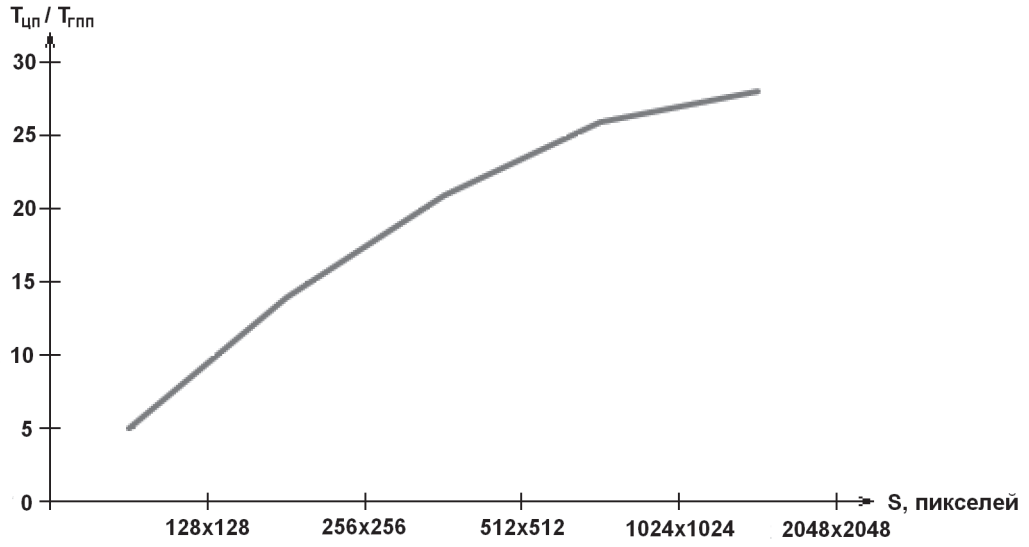

Рис. 5. Отношение времени выполнения обратного 2D ДВП на ЦП и ГПП.

Изображение на интерактивной доске в vAcademia состоит из двух слоев и представляет собой по сути динамическую текстуру. Изображение на доске обновляется генерацией статической текстуры комбинированным методом, т.к. с учетом условий задачи этот метод обеспечивает наилучшую производительность. Нижний слой это картинка, слайд или динамически обновляемое изображение инструмента «screen sharing», а верхний слой – растровое изображение векторного инструмента визуального комментирования. Результаты ускорения генерации текстуры этого векторного инструмента на ГПП приведены в [8].

10.7256/2305-6061.2013.01.10

7. Заключение

Описанные подходы, обобщенная математическая модель и программная архитектура генерации статических и динамических текстур на основе использования вычислительных ресурсов графических потоковых процессоров были успешно апробированы в образовательном виртуальном мире vAcademia, в том числе реализована генерация динамических текстур с непредопределенным содержимым для инструмента «screen sharing». Представленная математическая модель позволяет оценить различные подходы к генерации текстур и выбрать наиболее оптимальный.

Список литературы:

1. *М.Н. Морозов, А. В. Герасимов, М.Н. Курдюмова*. Системы совместной учебной деятельности на основе компьютерных сетей // Образовательные технологии и общество. 2009. Т. 12. № 1. С. 310-324.
2. *Д. А. Быстров*. Оптимизация сетевого трафика сообщений синхронизации объектов образовательного виртуального мира vAcademia // Образовательные технологии и общество. 2011. Т. 14. № 3. С. 439-456.
3. *Ares Lagae, Peter Vangorp, Toon Lenaerts, Philip Dutre*. Procedural isotropic stochastic textures by example // Computers & Graphics (Special issue on Procedural Methods in Computer Graphics). 2010. № 4. pp. 312-321.
4. *Gilberto Rosado*. Motion Blur as a Post-Processing Effect // GPU Gems 3. Addison-Wesley Professional, 2007. pp. 575-610.
5. *Xiaogang Jin, Chen Shaochun, Xiaoyang Mao*. Computer-Generated Marbling Textures: A GPU-based Design System // IEEE Computer Graphics and Applications, IEEE Computer Society. 2007. № 2. pp. 78-84.
6. URL: http://www.vacademia.com (дата обращения: 20.06.2012).
7. *Tien-Tsin Wong, Chi-Sing Leung, Pheng-Ann Heng*, and *Jianqing Wang*. Discrete Wavelet Transform on Consumer-Level Graphics Hardware // IEEE Trans. on Multimedia. 2007. No. 3. Vol. 9. pp. 668-673.
8. *Сморкалов А.Ю*. Поддержка операций рисования в системе обработки растровых изображений на потоковых процессорах // Сборник материалов Всероссийской научно-практической конференции "Информационные технологии в профессиональной деятельности и научной работе" / МарГТУ. Йошкар-Ола, 2011. С. 201-206.

Библиография:

1. М.Н. Морозов, А. В. Герасимов, М.Н. Курдюмова. Системы совместной учебной деятельности на основе компьютерных сетей // Образовательные технологии и общество. 2009. Т. 12. № 1. С. 310-324.

2. Д. А. Быстров. Оптимизация сетевого трафика сообщений синхронизации объектов образовательного виртуального мира vAcademia // Образовательные технологии и общество. 2011. Т. 14. № 3. С. 439-456.
3. Ares Lagae, Peter Vangorp, Toon Lenaerts, Philip Dutre. Procedural isotropic stochastic textures by example // Computers & Graphics (Special issue on Procedural Methods in Computer Graphics). 2010. № 4. pp. 312-321.
4. Gilberto Rosado. Motion Blur as a Post-Processing Effect // GPU Gems 3. Addison-Wesley Professional, 2007. pp. 575-610.
5. Xiaogang Jin, Chen Shaochun, Xiaoyang Mao. Computer-Generated Marbling Textures: A GPU-based Design System // IEEE Computer Graphics and Applications, IEEE Computer Society. 2007. № 2. pp. 78-84.
6. URL: http://www.vacademia.com (дата обращения: 20.06.2012).
7. Tien-Tsin Wong, Chi-Sing Leung, Pheng-Ann Heng, and Jianqing Wang. Discrete Wavelet Transform on Consumer-Level Graphics Hardware // IEEE Trans. on Multimedia. 2007. No. 3. Vol. 9. pp. 668-673.
8. C. Tenllado, J. Setoain, M. Prieto, L. Pinuel, F. Tirado. Parallel Implementation of the 2D Discrete Wavelet Transform on Graphics Processing Units: Filter Bank versus Lifting // IEEE Trans. Parallel And Distributed Systems. 2008. No 3. Vol. 19. pp. 299-310.
9. Сморкалов А.Ю. Реализация образовательных инструментов в виртуальных 3D-средах с использованием потоковых процессоров // Образовательные технологии и общество. 2011. Т. 14. № 3. С. 409-425.
10. Сморкалов А.Ю. Поддержка операций рисования в системе обработки растровых изображений на потоковых процессорах // Сборник материалов Всероссийской научно-практической конференции “Информационные технологии в профессиональной деятельности и научной работе” / МарГТУ. Йошкар-Ола, 2011. С. 201-206.

References (transliteration):

1. M.N. Morozov, A. V. Gerasimov, M.N. Kurdyumova. Sistemy sovmestnoy uchebnoy deyatel’nosti na osnove komp’yuternykh setey // Obrazovatel’nye tekhnologii i obshchestvo. 2009. T. 12. № 1. S. 310-324.
2. D. A. Bystrov. Optimizatsiya setevogo trafika soobshcheniy sinkhronizatsii ob’ektov obrazovatel’nogo virtual’nogo mira vAcademia // Obrazovatel’nye tekhnologii i obshchestvo. 2011. T. 14. № 3. S. 439-456.
3. Ares Lagae, Peter Vangorp, Toon Lenaerts, Philip Dutre. Procedural isotropic stochastic textures by example // Computers & Graphics (Special issue on Procedural Methods in Computer Graphics). 2010. № 4. pp. 312-321.
4. Gilberto Rosado. Motion Blur as a Post-Processing Effect // GPU Gems 3. Addison-Wesley Professional, 2007. pp. 575-610.
5. Xiaogang Jin, Chen Shaochun, Xiaoyang Mao. Computer-Generated Marbling Textures: A GPU-based Design System // IEEE Computer Graphics and Applications, IEEE Computer Society. 2007. № 2. pp. 78-84.

6. URL: http://www.vacademia.com (data obrashcheniya: 20.06.2012).
7. Tien-Tsin Wong, Chi-Sing Leung, Pheng-Ann Heng, and Jianqing Wang. Discrete Wavelet Transform on Consumer-Level Graphics Hardware // IEEE Trans. on Multimedia. 2007. No. 3. Vol. 9. pp. 668-673.
8. C. Tenllado, J. Setoain, M. Prieto, L. Pinuel, F. Tirado. Parallel Implementation of the 2D Discrete Wavelet Transform on Graphics Processing Units: Filter Bank versus Lifting // IEEE Trans. Parallel And Distributed Systems. 2008. No 3. Vol. 19. pp. 299-310.
9. Smorkalov A.Yu. Realizatsiya obrazovatel'nykh instrumentov v virtual'nykh 3D-sredakh s ispol'zovaniem potokovykh protsessorov // Obrazovatel'nye tekhnologii i obshchestvo. 2011. T. 14. № 3. S. 409-425.
10. Smorkalov A.Yu. Podderzhka operatsiy risovaniya v sisteme obrabotki rastrovykh izobrazheniy na potokovykh protsessorakh // Sbornik materialov Vserossiyskoy nauchno-prakticheskoy konferentsii «Informatsionnye tekhnologii v professional'noy deyatel'nosti i nauchnoy rabote» / MarGTU. Yoshkar-Ola, 2011. S. 201-206.