УДК 004.4
DOI 10.53403/9785951505071_2022_304

РЕАЛИЗАЦИЯ МЕХАНИЗМА ОТОБРАЖЕНИЯ РЕЗУЛЬТАТОВ МОДЕЛИРОВАНИЯ С ИСПОЛЬЗОВАНИЕМ ОБЪЕМНОГО РЕНДЕРИНГА В ПАРАЛЛЕЛЬНОМ РЕЖИМЕ

В. В Ломтев, В. В. Жирнов

Российский федеральный ядерный центр –
Всероссийский НИИ экспериментальной физики, Саров

В докладе представлен механизм отображения результатов моделирования с использованием объемного рендеринга [1] в параллельной системе постобработки ScientificView [2], предназначенной для визуализации результатов моделирования различных физических процессов и экспериментальных данных, представляемых в виде сеточной структуры или набора точек (частиц, молекул, кластеров).

Ключевые слова: ScientificView, объемный рендеринг, параллельный режим

Определение объемного рендеринга

Объемный рендеринг – технология компьютерной графики, которая позволяет представить распределение величины в объеме в виде облака пыли или тумана. В качестве исходных данных для отображения выступают результаты 3D-моделирования, представленные набором дискретных ячеек (сетка) с определенным значением величины в пределах каждой дискретной ячейки. Помимо цветовой интерпретации величины (как это делается в поверхностном полигональном случае), в данной технологии также используются соответствующие уровням величины значения полупрозрачности. Оперируя цветовой интерпретацией значений величины и соответствующими значениями полупрозрачности, пользователь может исключать определенные значения из анализа, или выделять их относительно других, как показано на рис. 1.

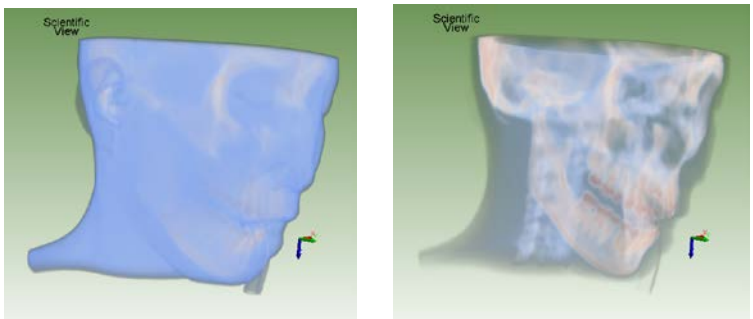

Рис. 1. Демонстрация работы технологии объемного рендеринга

Особенности реализации объемного рендеринга

Рендеринг объемных данных значительно отличается от рендеринга данных в классическом полигональном представлении. Современные видеокарты больше адаптированы к полигональной графике и способны обрабатывать десятки миллионов полигонов в секунду. И даже, когда речь идет

о смешивании цветов для поддержки полупрозрачных полигональных моделей, для каждого пикселя экрана идет обработка лишь небольшого числа фрагментов (некое значение цвета и глубины пикселя, которое претендует стать итоговым), в зависимости от количества используемых слоев глубины. При объемном рендеринге для каждого пикселя экрана необходимо обработать данные всех ячеек, находящиеся под позицией пикселя (все ячейки, которые проецируются в данный пиксель), что является ресурсозатратной задачей не только для видеокарты (в силу сильной ограниченности памяти), но и для центрального процессора, так как современные результаты моделирования могут содержать миллиарды ячеек. Поэтому, зачастую, исходные данные аппроксимируют на менее подробные сетки, состоящие из кубиков (вокселей), как показано на рис. 2. Обычно выбор размерности такой воксельной сетки в системах постпроцессинга производится пользователем в зависимости от ожидаемой степени соответствия исходным данным.

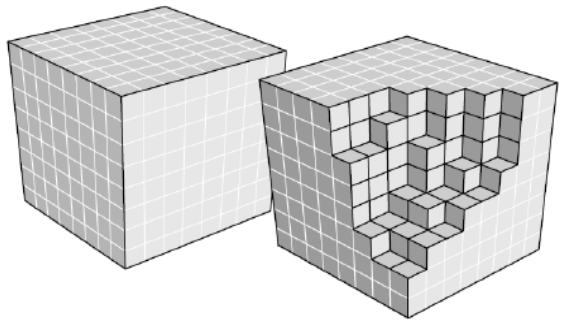

Рис. 2. Демонстрация воксельной сетки

Немаловажным фактором при отображении объемных данных является время формирования кадра. Даже на аппроксимированных сетках центральный процессор не сможет обеспечить скорость отображения, достаточную для интерактивного взаимодействия пользователя со сценой (далее – интерактивного взаимодействия), поскольку для каждого непустого пикселя необходимо найти и обработать все воксели, которые проецируются в данный пиксель. Использование ресурсов видеокарты позволит обеспечить высокую скорость формирования изображений в объемном представлении.

В параллельной системе постобработки ScientificView основным инструментом для визуализации как двумерных, так и трехмерных данных является графическая библиотека OpenGL [3]. Графическое ядро подсистемы визуализации ScientificView построено на базе API библиотеки OpenGL. При разработке универсальной концепции реализации объемного рендеринга решено было использовать только механизмы данной библиотеки и ее расширений, так как она является открытой, кроссплатформенной и поддерживается большинством из известных семейств видеокарт. К тому же, использование воксельного представления очень удобно для размещения объемных данных (матричная позиция вокселя и значение в нем) в памяти видеокарты. Такое представление хорошо согласуется с организацией 3D-текстур. Благодаря автоматической интерполяции, характерной для данного вида текстур, разработчик автоматически способен получить значение в любой точке объемных данных, даже если это положение не совпадает с центром вокселя.

В реализации объемного рендеринга используются механизмы соответствия цвета и коэффициента полупрозрачности значениям величины, распределенным по трехмерной сетке. В качестве таких механизмов удобно использовать одномерные текстуры. При этом для цветовой интерпретации используется текстура, заполняемая RGB значениями (слагаемые компоненты итогового цвета – красный, зеленый, синий), а для интерпретации коэффициента полупрозрачности используется текстура, заполняемая значениями компонента A (альфа). В системах визуализации задание вышеописанных соответствий является мощным инструментом для управления отображением исследуемых величин, так как пользователю предоставляется возможность скрыть определенные значения, сделав их полностью прозрачными (A = 0), или смешать цвета так, чтобы выделить определенные зоны данных, задавая контрастный цвет интересующим значениям с использованием коэффициента полупрозрачности наиболее близким к единице.

Процесс поиска вокселей, которые проецируются в определенный пиксель, заменяется на сбор значений величин, встречающихся при прохождении луча через 3D-текстуру в направлении от минимальной глубины к максимальной в позиции пикселя. При этом от шага прохождения вдоль луча зависит скорость формирования общего кадра. Шаг прохождения вдоль луча называют величиной сэмплирования, которая составляет характерный размер вокселя. Зачастую, с целью ускорения отображения при интерактивных взаимодействиях со сценой, в системах визуализации величину сэмплирования

увеличивают на определенный множитель. При этом прохождение по лучу проходит быстрее. По окончании интерактивного взаимодействия величина сэмплирования восстанавливается. Операции прохождения вдоль луча и смешения цветов реализуются во фрагментном шейдере, при этом в шейдер обязательно должны быть поданы в качестве uniform переменных: идентификатор 3D-текстуры объемных данных, идентификаторы 1D-текстур интерпретации цвета и компонента A, величина сэмплирования.

Концепция и алгоритм реализации объемного рендеринга в параллельной системе постобработки ScientificView

Таким образом, концептуально, формирование изображений с помощью объемного рендеринга можно разбить на три основных этапа:

– подготовка данных (не влияет на скорость интерактивного взаимодействия);

– предварительные операции отображения;

– трассировка лучей.

На этапе подготовки данных:

– исходные данные аппроксимируем на воксельную сетку, указанной пользователем размерности и вычисляем величину сэмплирования;

– при помощи API OpenGL на видеокарте выделяем память под 3D-текстуру и заполняем ее согласно распределению величины по воксельной сетке;

– при помощи API OpenGL на видеокарте выделяем память под 1D-текстуру цветовой интерпретации величины и заполняем ее согласно заданным пользователем уровням;

– при помощи API OpenGL на видеокарте выделяем память под 1D-текстуру соответствия величине компонента A и заполняем текстуру согласно заданным пользователем уровням.

На этапе предварительных операций отображения:

– находим минимальные и максимальные глубины для каждого пикселя. Для этого производим полигональное отображение (без цветовой интерпретации) данных для объемного анализа с включенным тестом глубины. В первый проход определяем минимальные глубины с функцией теста глубины GL_LESS, во второй проход определяем максимальные глубины с функцией теста глубины GL_GATHER. Результирующие буферы глубины сохраняем на видеоадаптере в виде текстур;

– включаем использование фрагментного шейдера.

При этом внутри фрагментного шейдера для каждого пикселя (этап трассировки лучей):

– определяем при помощи преобразований наблюдения модели и проекции 3D-координаты старта и окончания луча (по позиции пикселя и данным из текстур минимальных и максимальных глубин), при этом пропускаем пустые зоны, где значение минимальной глубины равно единице;

– проходим по лучу согласно величине сэмплирования, при этом каждую новую 3D-точку переводим в нормированные координаты текстур (в текстурах координаты нормируются в интервале от нуля до единицы);

– для каждой новой точки вдоль луча определяем величину из 3D-текстуры;

– согласно величине, определяем цвет и компонент A из соответствующих текстур, производим смешение цветов;

– при выходе луча из 3D-текстуры, заканчиваем обработку пикселя и устанавливаем для него цвет и компонент A согласно накопленным значениям.

В случае запуска системы ScientificView в режиме клиент-сервер, обработка данных и формирование изображений выполняются на стороне сервера. При этом сервер может работать как в последовательном, так и в параллельном многопроцессорном режиме с распределенной памятью. В последнем случае трассировка лучей проводится на специально выделенном управляющем процессоре, оснащенном видеокартой.

На управляющем процессоре проводится операция по сбору значений для итоговой воксельной сетки. С этой целью ячеечные величины исходных данных зачитываются с перехлестом на рабочих

процессорах. Это позволяет для каждого рабочего процессора определить воксели, которые не вносят вклад в итоговую сетку (в данные воксели попали все ячейки с перехлестного слоя), и воксели для которых формируется дополнительная информация для поиска итоговой величины (воксели, в которые попали как свои ячейки, так и ячейки с перехлестного слоя). В качестве дополнительной информации выступает определенное количество кортежей «расстояние от центра вокселя до центра ячейки – значение величины, соответствующее этому расстоянию».

После формирования воксельных представлений на рабочих процессорах, управляющий процессор запускает операцию попарного объединения среди рабочих, с динамическим распределением нагрузки. При этом в спорных вокселях происходит определение наиближайших кортежей из дополнительной информации. В результате итоговые значения для общей воксельной сетки концентрируются на управляющем процессоре, после чего запускается операция нахождения значений величины в спорных вокселях, исходя из определенного количества значений расстояний и соответствующих им значений величины.

С целью сохранения формы исходных данных, на управляющем процессоре запускается две операции объединения кадра для формирования минимальных и максимальных глубин. Данные глубины, согласно концепции, сохраняются в виде текстур на видеокарте для дальнейшего использования во фрагментном шейдере. В результате выполнения вышеописанных операций, на управляющем процессоре сосредотачивается вся необходимая информация для формирования представления с использованием объемного рендеринга.

Заключение

Возможность представления результатов моделирования при помощи технологии объемного рендеринга уже включена в последовательную версию постпроцессора ScientificView и доступна пользователям. В настоящее время автором ведутся работы по апробации механизма в параллельном режиме на реальных задачах пользователей.

Литература

1. Leven J., Corso J., Cohen J. Interactive visualization of unstructured grids using hierarchical 3D textures // IEEE 2002 Symposium on Volume Visualization and Graphics. − 2002. − Pp. 37–44.

2. Потехин А. Л., Логинов И. В., Лопаткин А. И., Ломтев В. В и др. Технология удаленной обработки результатов расчетов сверхбольшого объема // Журнал информационные технологии в проектировании и производстве. – 2017. № 2 (166). − С. 33–41.

3. Ричард С. Райт мл., Бенджамин Липчак. OpenGL. – М.: Суперкнига, 3-е издание. 2006.

IMPLEMENTATION OF THE SIMULATION RESULT REPRESENTATION MECHANISM USING VOLUME RENDERING IN PARALLEL MODE

V. V. Lomtev, V. V. Zhirnov

Russian Federal Nuclear Center –
All-Russian Scientific Research Institute of Experimental Physics, Sarov

The paper presents the mechanism of representing simulation results with the use of the volume rendering procedure [1] in the parallel postprocessing system ScientificView [2] developed for viewing various physical processes and experimental data, which are represented as a mesh structure, or a set of points (particles, molecules, clusters).

Key words: ScientificView, volume rendering, parallel mode.