The use of diagnosis and correction algorithms in the learning process allows to trace the connection between the set of "surface error" and a profound lack of the material understanding (diagnosis) and to classify errors and choose the relevant corrective strategy (correction).

In this article the possibility of programs creation for continuous diagnostics realization, combined with the process of the learning dialog and the implementation of the correction, that takes into account individual characteristics of the student, are investigated

Key words: *diagnostics, correction, teaching stimulus, teaching system, tutor.*

Отримано: 20.03.2012

УДК 004.946

С. И. Шаповалова, канд. техн. наук,
А. А. Форсюк, магистрант

Национальный технический университет Украины «КПИ», г. Киев

МОДЕЛИРОВАНИЕ ПОВЕДЕНИЯ ЖИДКОСТЕЙ МЕТОДОМ SPH НА БАЗЕ ВИДЕОАДАПТЕРОВ

В статье предложено программное решение для задачи симуляции динамики жидкостей методом SPH. Обосновано использование графического адаптера. Экспериментально доказаны преимущества предложенного решения.

Ключевые слова: *гетерогенные вычисления, CUDA, симуляция жидкости, SPH.*

Постановка проблемы. Моделирование поведения жидкости актуально для решения многих прикладных задач – от вычислительной гидромеханики до производства фильмов и компьютерных игр. Однако, эта проблема не может быть решена в реальном времени традиционными методами из-за аппаратных ограничений современных центральных процессоров. Поэтому создание программно-алгоритмической базы для реалистичной симуляции динамики жидкостей в реальном времени является актульным.

Анализ исследований. Наиболее распространёнными методами моделирования поведения жидкостей являются сеточные методы Эйлера, метод гидродинамики сглаженных частиц SPH (англ. smoothed particle hydrodynamics) [1], метод решёточных уравнений Больцмана [2], и методы, основанные на завихрениях [3]. Эти методы были разработаны исследователями вычислительной гидродинамики и позаимствованы для решения практических задач в индустрии компьютерной графики и спе-

цэффектов. Обладая наилучшим соотношением визуальной правдоподобности симуляции и простоты вычислений, метод SPH чаще остальных используется при разработке игр и визуальных эффектов в фильмах. Метод гидродинамики сглаженных частиц может эффективно симулировать гидродинамические потоки, однако это не единственное его применение. Основанная на частицах «природа» SPH делает его идеальным выбором для объединения с обработчиком гравитации и симуляций движения космических тел в астрофизике. Среди недостатков метода можно выделить тенденцию уменьшать («смазывать» от англ. smear out) ударные волны и контактные разрывы, что снижает физическую достоверность симуляции.

Метод SPH заключается в делении жидкости на дискретные элементы, называемые частицами, и индивидуальной обработке каждой из них. Поскольку суть метода заключается в независимой обработке дискретных частиц, большую выгоду можно извлечь из применения массово-параллельных вычислений. Требуемая визуальная правдоподобность может быть достигнута за счёт использования большого количества частиц. Однако существующие реализации метода SPH не могут работать с большими массивами частиц в реальном времени из-за аппаратных ограничений вычислительных мощностей современных центральных процессоров. В работе Матиаса Мюллера (Matthias Möller) [1], в которой использовались вычисления на центральном процессоре, была реализована симуляция 3000 частиц со скоростью 5 кадров в секунду. Подобный результат не позволяет достичь необходимой визуально правдоподобной симуляции в реальном времени.

Исследователи корпорации Advanced Micro Devices (AMD, США) смогли достичь 17 кадров в секунду при симуляции 49000 частиц, используя гетерогенные вычисления с применением видеоадаптера [4]. Результат демонстрирует ефективность использования видеоадаптера, но полученная скорость значительно ниже скорости реального времени, которая составляет 30 кадров в секунду. Группе исследователей из Пекинского института [5] использованием нескольких видеоадаптеров удалось достичь симуляции 60000 частиц со скоростью 57 кадров в секунду. Однако предложенная система из нескольких видеоадаптеров редко встречается у пользователей, поэтому предложенное решение не применимо в потребительских продуктах.

Цель – предложить реализацию метода SPH, поддерживающую обработку количества частиц четвёртого порядка в реальном времени, на основе технологии nVidia CUDA.

Метод SPH

Каждая дискретная частица, используемая в методе SPH, представляется следующими физическими характеристиками: положение в пространстве, скорость, плотность, вязкость, силы, действующие со

стороны других частиц и иных обьектов симуляции. Значение любой физической величины A в точке r, задаётся формулой [1]:

$$A(r) = \sum_j m_j \frac{A_j}{\rho_j} W\left(\left|r - r_j\right|, h\right), \tag{1}$$

где m_j — масса j-й частицы, A_j — значение величины A для j-й частицы, c_j — плотность, связанная с j-й частицей, W — функция ядра, h — пространственное расстояние.

Например, плотность i-й частицы может быть представлена как:

$$\rho_i = \rho(r_i) = \sum_j m_j \frac{\rho_j}{\rho_j} W\left(\left|r_i - r_j\right|, h\right) = \sum_j m_j W\left(\left|r_i - r_j\right|, h\right), \tag{2}$$

где суммирование j включает все частицы в симуляции.

Пространственное расстояние, известное как «длина сглаживания», является расстоянием, на котором свойства частицы «сглаживаются» функцией ядра. Таким образом, каждая физическая величина каждой частицы может быть получена путем суммирования соответствующих величин всех частиц, которые находятся в пределах длины сглаживания. Влияние каждой частицы на свойства оценивается в соответствии с её плотностью и расстоянием до интересующей частицы и определяется функцией ядра W. В качестве функции ядра, как правило, используют функцию Гаусса или кубический сплайн. Функция сглаживания, использующая кубический сплайн, в общем виде определяется следующим образом [6]:

$$W(r,h) = \frac{\sigma}{h^v} \begin{cases} 1 - \frac{3}{2}q^2 + \frac{3}{4}q^3, & 0 \le q \le 1; \\ \frac{1}{4}(2-q)^3, & 1 \le q \le 2; \\ 0, & q > 2, \end{cases} \tag{3}$$

где v — размерность пространства, σ — нормализационная константа, значения которой — $\frac{2}{3}, \frac{10}{7\pi}, \frac{1}{\pi}$ для одно-, двух- и трёхмерного пространства соответственно, $q = \frac{r}{h}$.

Значение функции $W(r,h)$ равно нулю для частиц, находящихся дальше, чем две сглаженные длины. Это позволяет экономить вычислительные ресурсы, исключая расчёты относительно малого влияния отдаленных частиц, но негативно влияет на физическую достоверность симуляции.

Ядро сглаживания на основе функции Гаусса для трёхмерного пространства определяется следующим образом [6]:

$$W(r,h) = \frac{1}{\pi \frac{3}{2} h^3} \left(\frac{5}{2} - r^2 \right) e^{-\frac{r^2}{h^2}}. \quad (4)$$

В этом случае учитывается даже небольшое влияние вершин, удалённых более чем на две сглаженные длины, однако требуется гораздо больше вычислений. Использование нормального распределения Гаусса позволяет достичь наиболее физически достоверных результатов.

Классическая реализация метода SPH использует фиксированную длину сглаживания. Однако это не позволяет достичь максимально визуально правдоподобной симуляции. Назначая каждой частице её собственную длину сглаживания, изменяющуюся со временем, разрешающая способность симуляции может автоматически адаптироваться к локальным условиям. Например, в очень плотной области длина сглаживания может быть установлена относительно короткой, приводя к высокому пространственному разрешению. И, наоборот, в областях с малой плотностью длина сглаживания может быть увеличена, тем самым оптимизируя вычисления.

Технология CUDA

CUDA (англ. Compute Unified Device Architecture) [7] — программно-аппаратная архитектура, позволяющая производить вычисления на базе графических процессоров NVIDIA, поддерживающих технологию GPGPU (произвольных вычислений на видеокартах). Впервые появилась на рынке с выходом чипа NVIDIA восьмого поколения — G80 и присутствует во всех последующих сериях графических чипов, которые используются в семействах ускорителей GeForce, Quadro и Tesla.

Для эффективного использования вычислительных возможностей GPU нужно учитывать аппаратные особенности архитектуры видеочипов nVidia. GPU состоит из нескольких кластеров текстурных блоков (Texture Processing Cluster). Каждый кластер включает укрупнённый блок текстурных выборок и два-три потоковых мультипроцессора. Каждый из последних состоит из восьми вычислительных устройств и двух суперфункциональных блоков. Все инструкции выполняются по принципу SIMD, когда одна инструкция применяется ко всем потокам в warp. Warp представляет собой группу из 32 потоков — минимальный объём данных, обрабатываемых мультипроцессорами. Этот способ выполнения назвали SIMT (single instruction multiple threads — одна инструкция и много потоков). Каждый из мультипроцессоров имеет специальную разделяемую память, которую можно явно использовать для обмена информацией между потоками одного блока.

Программа компилируется при помощи поставляемого NVIDIA компилятора, который генерирует код и для CPU, и для GPU. При исполнении программы, центральный процессор выполняет свои порции кода, а GPU выполняет CUDA код с наиболее тяжелыми параллельными вычислениями. Эта часть, предназначенная для GPU, называется ядром (kernel).

Видеочип получает ядро и создает копии для каждого элемента данных. Эти копии называются потоками (thread). Поток содержит счётчик, регистры и состояние. Для больших объёмов данных, таких как обработка изображений, запускаются миллионы потоков. Потоки выполняются warp'ами. Warp'ам назначается исполнение на определенных потоковых мультипроцессорах. Каждый мультипроцессор состоит из восьми ядер — потоковых процессоров, которые выполняют одну инструкцию за один такт. Для исполнения одного 32-поточного warp'а требуется четыре такта работы мультипроцессора. В отличии от центрального процессора мультипроцессор GPU выполняет переключение между программами без потери тактов.

Реализация метода SPH

Вычисления метода SPH представляют собой последовательность из следующий этапов: поиск соседних частиц, расчёт сил согласно формулам метода, отрисовка результата.

Суть этапа поиска соседних частиц заключается в создании вспомогательных структур, с помощью которых можно обеспечить быстрый доступ ко всем соседям определенной частицы. Для реализации используется техника «широкой фазы» [8]. Для широкой фазы используется Spatial Hashing — разделение всего пространства на множество ячеек. В случае, когда длина сглаживания равна размеру ячейки, применение этой техники не влияет на достоверность симуляции. Хэш каждой отдельно взятой частицы равен индексу ячейки пространства, в которую она попадает. Техника оперирует двумя дополнительными таблицами: таблица хешей частиц (каждый елемент содержит индекс частицы и её пространственный хеш) и таблица адресов (каждий елемент содержит индекс ячейки пространства, индекс первой частицы, входящей в ячейку, в таблице хешей, количество частиц).

Работа метода разделяется на 3 промежуточных шага: вычисление таблицы хэшей, сортировка этой таблицы со значением хэша в качестве ключа и построение таблицы адресов. Для сортировки использовалась структура данных device_vector и реализованый в ней алгоритм Fast Radix Sort, оптимизированый для выполнения на GPU, из библиотеки thrust. Обе таблицы во время работы хранятся в текстурной памяти видеоадаптера. Обращения к ним производятся только из кода, выполняемого в ядрах CUDA.

На этапе расчёта сил производятся вычисления плотности, силы давления и вязкости для каждой из частиц.

Плотность, связанная с частицей, вычисляется по формуле (2), используя кубический сплайн в качестве функции сглаживания, для экономии вычислительных ресурсов. При расчётах, во внимание принимаются соседние частицы, определённые с помощью техники широкой фазы, в результате чего отсекаются частицы, находящиеся дальше двух сглаженных длин.

Сила давления со стороны других частиц определяется модифицированным уравнением состояния идеального газа:

$$P_i = k\left(\rho_i - \rho_0\right), \tag{5}$$

где k — газовая константа, зависящая от температуры, ρ_0 — плотность спокойствия.

Выбранное значение k определяет скорость распространения упругих волн в среде жидкости. Сила давления способствует перемещению частиц в зоны низкой плотности из зон высокой.

Сила вязкости, ассоциированная с частицей, направлена на усреднение скоростей соседних частиц. Значение вязкости зависит от разницы скоростей текущей и соседних частиц и задаётся следующим образом:

$$f_i^{\text{viscosity}} = \mu \sum_j m_j \frac{v_j - v_i}{\rho_j} \nabla^2 W\left(r_i - r_j, h\right), \tag{6}$$

где μ — константа вязкости.

Чем выше значение константы вязкости, тем в большей степени будут усредняться скорости, в результате чего будет симулирована более вязкая жидкость.

Таким образом, можно получить следующее положение i-й частицы. Сначала вычисляется скорость частицы:

$$v_i = v_i + \Delta t\left(P_i + f_i^{\text{viscosity}}\right). \tag{7}$$

Затем — новое положение частицы в пространстве:

$$r_i = r_i + \Delta t v_i \tag{8}$$

Все вычисления метода для одной частицы выполняются на одном потоке. На каждой итерации работы метода SPH на видеоадаптере создаётся массив потоков чисельностью, равной чисельности частиц симуляции, сгруппированный в блоки согласно модели выполнения CUDA. Чисельность потоков в блоке ограничивается аппаратными возможностями видеоадаптера.

Рендеринг симуляции производится с помощью технологии Direct3D 10.1, которая обеспечивает полное аппаратное ускорение отрисовки. Большой прирост производительности даёт использование механизма интеграции технологий CUDA и Direct3D [7], суть которо-

го заключается в возможности использования в CUDA-вычислениях тех же ресурсов, которые будут использованы видеоадаптером для визуализации. Поскольку вычисления всех сил, скоростей и позиций частиц выполняются над данными, которые физически находятся в памяти видеоадаптера, то использование интеграции просчётов и визуализации позволяет исключить любой обмен данными между видеоадаптером и центральным процессором во время работы. В таблице 1 приводится модель хранения данных в памяти видеоадаптера.

Таблица 1

Модель хранения данных

Данные	**Используемая память**
Массив частиц	Массив обьектов, глобальная память. Каждый елемент хранит позицию и скорость частицы.
Положение частиц на предыдущей итерации	Четырёхкомпонентная текстура, текстурная память. Компоненты R, G, B хранят координаты x, y, z частицы. Компонент A не используется.
Скорости частиц на предыдущей итерации	Четырёхкомпонентная текстура, текстурная память. Компоненты R, G, B хранят компоненты x, y, z вектора скорости частицы. Компонент A не используется.
Таблица хешей техники широкой фазы	Двухкомпонентная текстура, текстурная память. Компонент R хранит уникальный идентификатор частицы, G – пространственный хеш частицы. Количество элементов равно количеству частиц симуляции.
Таблица адресов техники широкой фазы	Четырёхкомпонентная текстура, текстурная память. Компонент R хранит индекс ячейки пространства, G – индекс первой частицы в таблице хешей, B – количество частиц в таблице хешей. Компонент A не используется. Количество елементов равно количеству ячеек пространства.
Карта нормалей ограничивающей геометрии	Четырёхкомпонентная текстура, текстурная память.
Карта высот ограничивающей геометрии	Четырёхкомпонентная текстура, текстурная память.

Входными данными для визуализации является массив позиций частиц, каждый элемент которого представляет собой трёхмерный вектор. Эти данные группируются в, так называемый, вершинный буфер. Для

генерации геометрии сфер, которые представляют частицы, используется геометрический шейдер. Генерация геометрии во время визуализации позволяет экономить память видеоадаптера и его процессорное время, поскольку выполняется только один вызов графического драйвера.

Тестирование

Тестирование решения проводится с применением видеоадаптеров GeForce 260 GTX и GeForce 570 GTX. Тестовый сценарий симулирует падение массы жидкости в форме сферы на ландшафт. Ландшафт строится из карт высот и нормалей, которые используются для определения столкновений с частицами жидкости.

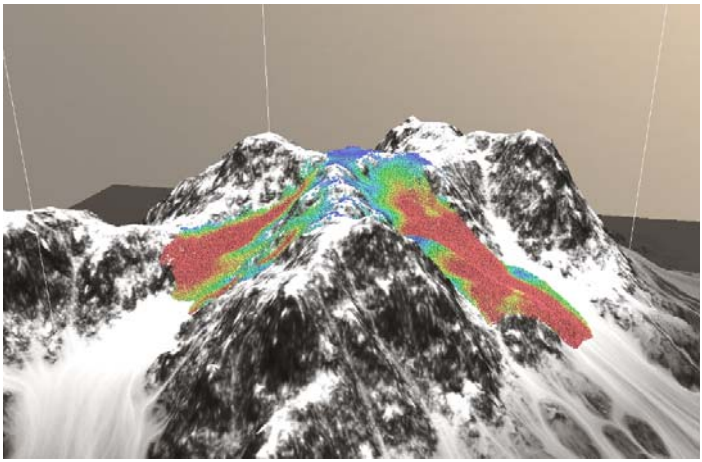

Рис. 1. *Визуализация симуляции после столкновения*

В тестовой симуляции установлены следующие значения параметров метода SPH:

Таблица 2

Парамметры симуляции

Имя парамметра	**Значение**
Шаг времени, Δt	0.0005
Плотность спокойствия, ρ_0	1000.0
Константа вязкости, μ	1.0
Масса частиц, m	0.0008
Длина сглаживания, h	0.0161527

Критерием измерения производительности выбрано количество итераций метода SPH в секунду. Подсчёт проводится с помощью предлагаемого технологией CUDA механизма событий и таймеров [7]. На рисунке 2 приведены результаты тестирования предложенной реализации и аналогичных решений других исследователей.

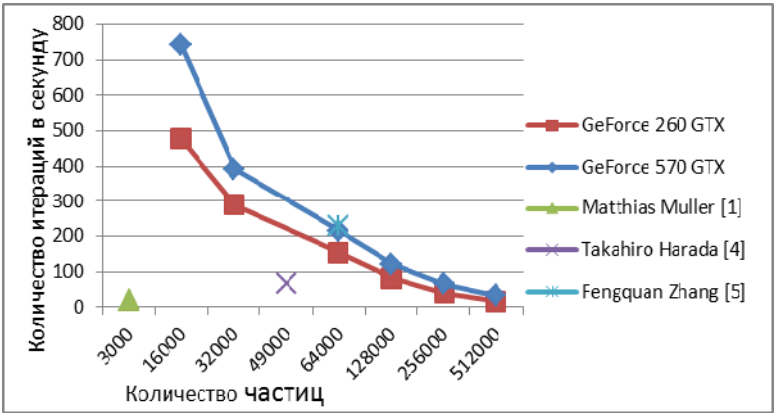

Рис. 2. *Результаты тестирования*

Выводы

1. Проанализированы программно-алгоритмические реализации метода SPH для симуляции поведения жидкости в приложениях реального времени. Определены недостатки предыдущих решений.
2. Предложена реализация метода SPH с применением технологии CUDA. Экспериментально доказано, что предложенное решение поддерживает обработку количества частиц 4-го порядка в реальном времени.
3. Проведен сравнительный анализ эффективности симуляции с использованием предложенного решения на базе видеоадаптеров GeForce 260 GTX и GeForce 570 GTX. Доказано, что в случае симуляции динамики большого количества частиц, предложенное решение превосходит существующие в скорости без потери визуальной правдоподобности.

Список использованной литературы:

1. Möller M. Particle-based fluid simulation for interactive applications / M. Möller, D. Charypar, M. Gross // Proceedings of the 2003 ACM SIGGRAPH / Eurographics symposium on Computer animation. — Aire-la-Ville, 2003. — P. 154–159.
2. Sauro Succi. The Lattice Boltzmann Equation for Fluid Dynamics and Beyond / Succi Sauro. — Oxford University Press, 2001. — 288 p.
3. Трошкин О. В. Отрывной вихрь в потоке вязкой жидкости / О. В. Трошкин // Журнал вычислительной математики и математической физики. – 1992. — № 32. — С. 329–331
4. Harada Takahiro. Heterogeneous particle-based simulation / Takahiro Harada // SIGGRAPH Asia 2011 Sketches. — New York, 2011. — P. 19.

5. Zhang Fengquan. A SPH-based method for interactive fluids simulation on the multi-GPU / Fengquan Zhang, Lei Hu, Jiawen Wu, Xukun Shen // Proceedings of the 10th International Conference on Virtual Reality Continuum and Its Applications in Industry. — New York, 2011. — P. 423–426
6. Monaghan J. J. Smoothed particle hydrodynamics / J. J. Monaghan // Annual Review of Astronomy and Astrophysics. — Clayton, 1992. — P. 543–574.
7. Kirk David B. Programming Massively Parallel Processors A Hands-on approach / David B. Kirk, Wen-mei W. Hwu. — Burlington : Morgan Kaufmann, 2010. — 258 p.
8. Scott Le Grand. Broad-Phase Collision Detection with CUDA / Scott Le Grand // GPU Gems 3. — Boston : Addison-Wesley Professional, 2007. — P. 697–721.

In this work software solution for fluid dynamics simulation through SPH-method is proposed. Hardware choise based graphics adapter is justified. Solution preferences are proved by experiment.

Key words: *heterogeneous computing, fluid simulation, CUDA, SPH.*

Отримано: 24.03.2012