НАУЧНОЕ ИЗДАНИЕ МГТУ ИМ. Н.Э. БАУМАНА
НАУКА и ОБРАЗОВАНИЕ
Эл № ФС 77 - 48211. Государственная регистрация №0421200025. ISSN 1994-0408
электронный научно-технический журнал

Реализация метода SPH на CUDA для моделирования несжимаемых жидкостей

 # 07, июль 2012
DOI: 10.7463/0712.0423582
Суравикин А. Ю.
УДК 004.942

Россия, ФГБОУ ВПО «Омский Государственный Университет им. Ф.М. Достоевского»
tizona_del_cid@rambler.ru

1. Введение

При изучении естественных течений, при разработке плотин, водных судов, гидроэлектростанций и других сооружений необходим точный расчет воздействия как окружающей среды и искусственных сооружений на течение жидкости, так и течения жидкости на сооружения и среду. Для оценки такого взаимодействия строится модель движения несжимаемой жидкости. Прежде чем начинать строительство дорогостоящего сооружения или устройства, можно оценить его надежность в различных условиях. Несмотря на то, что сейчас существует и развивается множество алгоритмов расчета гидродинамических нагрузок, часто требуется оценить надежность устройств и сооружений в широком диапазоне условий. В таких случаях строят модель сооружения и тестируют ее с помощью построения модели различных течений жидкостей.

Моделирование жидкости связано с решением нелинейных дифференциальных уравнений, которые в общем случае не имеют аналитического решения. Для того чтобы найти решение, используются численные методы, реализации которых выполняются на высокопроизводительных вычислительных системах.

В работе используется метод сглаженных частиц (Smoothed Particles Hydrodynamics, SPH), который является бессеточным лагранжевым численным методом. Бессеточные методы позволяют проводить расчеты течений с сильными деформациями границ расчетной области,

которые допускают изменение связности области расчета и перехлест границ области расчета. Для реализации указанных методов не требуется информация о связях между узлами, что позволяет избежать проблем, связанных с построением сетки, а также с необходимостью отслеживать межузловые связи.

Работа посвящена моделированию несжимаемой жидкости методом сглаженных частиц с помощью CUDA — программно-аппаратной архитектуры, позволяющей производить параллельные вычисления с использованием графических процессоров NVIDIA. Модификация указанного метода, основанная на модели несжимаемой жидкости «предиктор-корректор» (PCISPH) [9], позволила реализовать на графическом процессоре систему моделирования течения жидкостей. Было проведено сравнение результатов моделирования с экспериментальными результатами. Для оценки эффективности использования CUDA было проведено сравнение производительности метода, реализованного на центральном процессоре AMD Phenom X4 9850 и графическом процессоре NVIDIA GeForce GTX 470 в двумерном случае.

2. Уравнения Навье–Стокса

Уравнения Навье–Стокса — это система дифференциальных уравнений, которая описывает течение жидкостей. Система состоит из уравнения неразрывности и уравнения движения сплошной среды [10]. Уравнение неразрывности для несжимающихся жидкостей выглядит следующим образом:

$$\nabla \cdot \mathbf{v} = 0,$$

где $\nabla \cdot \mathbf{v}$ — дивергенция поля скорости, то есть $\nabla \cdot \mathbf{v} = \frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z}$.

Уравнение движения для вязкой жидкости имеет вид

$$\frac{\partial \mathbf{v}}{\partial t} = -\frac{1}{\rho}\nabla p + \nabla \cdot (\nu \nabla \mathbf{v}) + \mathbf{F}_{ext}.$$

Здесь ρ — плотность, **v** — скорость жидкости в некоторой точке, p — давление, ν — коэффициент кинематической вязкости, $\mathbf{F}_{ext}$ — внешние силы, приведенные к единице массы.

3. Аппроксимация функций

В основе метода сглаженных частиц лежит аппроксимация поля набором частиц. Чтобы обеспечить такую аппроксимацию, используется интегральная форма записи, которая

«сглаживает» области между частицами с помощью специальной функции-ядра. Определим некоторую область сглаживания $\Omega^{(\mathbf{r})} = \{\mathbf{r}' : |\mathbf{r} - \mathbf{r}'| < kh\}$, где *kh* — длина сглаживания. Параметр *k* зависит от функции-ядра, а *h* выбирается в зависимости от радиуса частицы. Тогда аппроксимация функции $f(\mathbf{r})$ будет выглядеть так:

$$\langle f(\mathbf{r})\rangle = \int\limits_{\Omega^{(\mathbf{r})}} f(\mathbf{r}')W(\mathbf{r} - \mathbf{r}', h)\, d\mathbf{r}',$$

где *W* — функция-ядро, которая обладает свойствами:

$$\int\limits_{\Omega^{(\mathbf{r})}} W(\mathbf{r} - \mathbf{r}', h)d\mathbf{r}' = 1,$$

$$\lim_{h\to 0} W(\mathbf{r} - \mathbf{r}', h) = \delta(|\mathbf{r} - \mathbf{r}'|),$$

$$W(\mathbf{r} - \mathbf{r}', h) = 0, r \notin \Omega^{(\mathbf{r})},$$

где $\delta(\mathrm{r})$ — дельта-функция Дирака. Функция-ядро также устанавливает параметр *k* в формуле длины сглаживания *kh*. Пример функции-ядра с $k = 2$ [6]:

$$W_{Liu}(R) = k^D_{Liu}\cdot\begin{cases} \frac{2}{3} - \frac{9}{8}R^2 + \frac{19}{24}R^3 - \frac{5}{32}R^4 & \text{при} \quad 0 \le R \le 2, \\ 0 & \text{иначе,} \end{cases} \quad \text{где } R = \frac{|\mathbf{r} - \mathbf{r}'|}{h}.$$

Коэффициент k^D_{Liu} зависит от размерности эксперимента *D* и длины сглаживания:

$$k^2_{Liu} = \frac{15}{7\pi h^2},\ k^3_{Liu} = \frac{305}{208\pi h^3}.$$

Заметим, что функция $f(\mathbf{r})$ известна только в некоторых точках *i*, к каждой из которой соотнесен некоторый объем:

$$V_i = \frac{m_i}{\rho_i}.$$

Поэтому можно перейти от непрерывной вычислительной области к дискретной, заменив интеграл по области сглаживания $\Omega^{(\mathbf{r})}$ на сумму по всем *N* частицам, которые находятся в области $\Omega^{(\mathbf{r})}$:

$$\langle f(\mathbf{r}_i)\rangle \approx \sum_j^{j\in\Omega_i} \frac{m_j}{\rho_j} f(\mathbf{r}_j)W(\mathbf{r}_i - \mathbf{r}_j, h),$$

где $\Omega_i = \Omega^{(\mathbf{r}_i)}$. Дифференциал функции вычисляется аналогично:

$$\langle \nabla \cdot f(\mathbf{r})\rangle \approx -\sum_j^{j\in\Omega_i} \frac{m_j}{\rho_j} f(\mathbf{r}_j)\nabla_i W_{ij}, \quad \text{где} \quad \nabla_i W_{ij} = \frac{\partial W(r,h)}{\partial r}\cdot\frac{\mathbf{r}_{ij}}{r}, \mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j, r = |\mathbf{r}_{ij}|.$$

4. Моделирование несжимаемой жидкости

При моделировании жидкостей методом SPH требуется на каждом временном шаге моделирования вычислять силу воздействия F_i на каждую частицу *i*:

$$\mathbf{F}_i = \mathbf{F}^{pres} + \mathbf{F}^{visc} + \mathbf{F}^{tens} + m\mathbf{g},$$

где $\mathbf{F}^{pres}$ — сила давления, $\mathbf{F}^{visc}$ — сила вязкости, $\mathbf{F}^{tens}$ — сила поверхностного натяжения, $m\mathbf{g}$ — сила гравитации. При моделировании экспериментов больших масштабов (> 10 см) силой поверхностного натяжения можно пренебречь.

После вычисления $\mathbf{F}_i$ производится численное интегрирование, при котором обновляются векторы скоростей (за счет ускорения, приданного силой воздействия) и позиций частиц.

Для вычисления сил взаимодействия между частицами требуется обеспечивать симметрию воздействия, то есть для всех частиц *i* и *j* должно выполняться равенство $\mathbf{F}_{ij} = -\mathbf{F}_{ji}$. Сила давления вычисляется, например, по формуле

$$F_i^{pres} = -m_i \sum_j \frac{p_i + p_j}{\rho_i \rho_j} \nabla_i W_{ij}, \tag{1}$$

где *j* — соседи частицы *i*, p_i, p_j, — давление частиц, которое вычисляется в зависимости от типа жидкости. Плотность частицы ρ_i вычисляется с помощью аппроксимации ядром функции $f(\mathbf{r})=\rho(\mathbf{r})$:

$$\rho_i \approx \langle \rho(\mathbf{r}_i) \rangle = \sum_j^N \frac{m_j}{\rho_j} \rho_j W(\mathbf{r}_i - \mathbf{r}_j, h) = \sum_j^N m_j W(\mathbf{r}_i - \mathbf{r}_j, h). \tag{2}$$

Вычисление силы вязкости допускает несколько подходов в вычислении. Простейшим с точки зрения вычислительных затрат является вычисление аппроксимации оператора Лапласа:

$$F_i^{visc} \approx \nu \nabla^2 v \approx m_i \sum_j \nu_j (\mathbf{v}_j - \mathbf{v}_i) \nabla_i^2 W_{ij}.$$

Существует несколько подходов при моделировании несжимаемой жидкости, которые отличаются, в основном, способом вычисления сил давления. Один из таких подходов — «слабо-сжимаемый» метод сглаженных частиц (WCSPH, Weakly Compressible SPH) [11]. Он основан на вычислении давления для частицы *i* по формуле:

$$p_i = \begin{cases} \frac{\rho_0 c_0}{\gamma} \left(\left(\frac{\rho_i}{\rho_0} \right)^\gamma - 1 \right), & \rho_i > \rho_0, \\ 0, & \rho_i \le \rho_0, \end{cases}$$

где $\gamma = 7$, c_0 — коэффициент жесткости, равный максимально возможной скорости частицы в эксперименте, ρ_0 — исходная плотность жидкости, ρ_i — плотность i-й частицы. Заметим, что если в некоторой точке происходит разрежение среды, то есть $\rho_i < \rho_0$, то вычисленное давление станет отрицательным. В таком случае для этой точке устанавливаем давление, равное нулю.

Однако метод WCSPH обладает существенными недостатками: для моделирования течений воды с большими деформациями с достаточной точностью потребуется очень малый шаг по времени. Так, скорость звука в воде c = 1435 м/с, то в общем случае коэффициент c_0 также должен быть равен 1435 м/с, поэтому в соответствии с условием Куранта для моделирования с точностью в 0,01 м потребуется шаг времени $\Delta t = 2{,}8 \times 10^{-6}$ с. Еще одним недостатком является неточное измерение давления жидкости [8].

Метод PCISPH (Predictive—Corrective Incompressible SPH) [9] обеспечивает более точное измерение давления и гарантирует несжимаемость при более высоких значениях шага времени Δt за счет итерационного процесса вычисления силы давления. В данном методе для каждой частицы сначала определяется значение всех сил, кроме давления. Затем сила давления определяется итерационно, в зависимости от предсказанного набора позиций частиц. При этом производится проверка максимальной ошибки плотности частиц $\rho_i^{err} = \rho_i - \rho_0$ для текущего размещения. Схема алгоритма указана на рис. 1. Слева от каждого шага показаны массивы, из которых производится чтение данных, а справа — массивы, в которые производится запись на данном шаге.

Каждый шаг алгоритма выполняется параллельно, при этом запускается N независимых потоков, где N — число частиц. Между шагами выполняется барьерная синхронизация. В начале алгоритма происходит заполнение массива соседей **neighbors** и инициализация значений ошибок плотности, давления и сил давления:

$$\rho_i^{err} = 0, p_i = 0, \mathbf{F}_i^{pres} = (0, 0, 0).$$

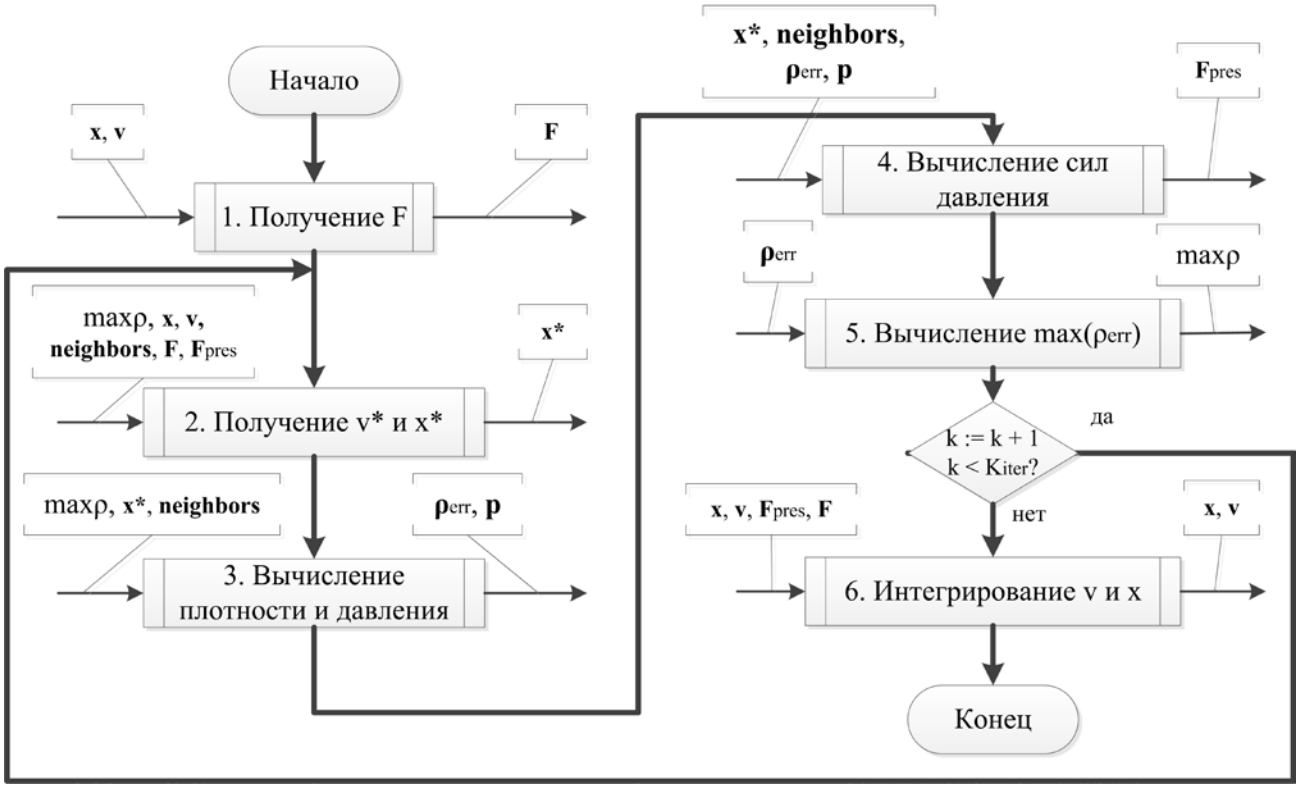

Рис. 1. Схема алгоритма PCISPH.

Сила давления вычисляется по формуле (1). Коэффициент δ вычисляется по формуле

$$\delta = \frac{-1}{\beta\left(-\sum_j \nabla W_{ij} \cdot \sum_j \nabla W_{ij} - \sum_j (\nabla W_{ij} \cdot \nabla W_{ij})\right)},$$

где $\beta = \Delta t^2 m^2 \frac{2}{\rho_0^2}$,

Δt — временной шаг.

5. Реализация алгоритма SPH на архитектуре CUDA.

При реализации метода сглаженных частиц на CUDA для достижения высокой производительности требуется учитывать особенности архитектуры графического процессора (Graphics Processing Unit (GPU)). Рассмотрим основные этапы вычислений:

1. Хеширование позиций частиц.

2. Формирование массива соседей для каждой частицы.

3. Вычисление сил, действующих на частицы.

4. Обработка граничных условий.

5.1. Хеширование позиций частиц

Метод сглаженных частиц предполагает вычисление взаимодействия частиц, которые находятся ближе чем на расстоянии kh друг от друга, то есть соседние частицы. Чтобы определить множество соседей частицы i, можно перебирать все частицы и определять расстояние между всеми частицами. Очевидно, что трудоемкость такой задачи $O(N^2)$, где N — количество частиц. С учетом того, что эти расстояния будут меняться на каждом шаге моделирования, определение соседей будет «узким местом» вычислений.

Для снижения временных затрат на вычисление соседей будем использовать хеширование. Пространство моделирования представляется в виде однородной циклической сетки, и каждой частице ставится в соответствие номер ячейки в этой сетке. Тогда трудоемкость снижается до $O(3^s c_n N)$, где c_n — среднее количество частиц в одной ячейке, s — размерность пространства.

Значение хеш-функции позиций частиц вычисляется параллельно для каждой частицы и является результатом вычисления выражений:

$$hash(\mathbf{r}) = h^z d^x d^y + h^y d^x + h^x,$$

где $h^\alpha = \frac{r_i^\alpha - r_{min}^\alpha}{cell} \mod d^\alpha,$

α — одна из пространственных координат (x, y или z), r_{min} — минимально возможные координаты частицы в данном эксперименте, $\mathbf{r}_i$ — позиция i-й частицы, $cell$ — размер ячейки, d — целочисленный вектор, содержащий размеры хеш-таблицы по трем осям, причем каждый его компонент d^α равен некоторой степени 2.

5.2. Формирование массива соседей для каждой частицы

Частицы, имеющие общее значение хеш-функции $hash(\mathbf{r})$, группируются последовательно. Также формируются массивы, содержащие для каждой ячейки количество частиц и адрес в новом массиве частиц, по которому находится первая частица с заданным значением $hash(\mathbf{r})$. Эти данные можно использовать для перебора соседей в каждом ядре CUDA, но в этом случае придется для каждого соседа проверять расстояние до исходной частицы. Чтобы избежать лишних вычислений и обращений к памяти, соседи для каждой частицы рассчитываются в начале каждого шага моделирования после формирования массивов значений хеш-функции, на этом же этапе проверяется расстояние до частицы и адрес соседа записывается в память GPU. Требование пересчитывать соседей связано с тем, что частицы

постоянно перемещаются, и множество соседей будет отличаться на различных шагах моделирования. Соседей можно пересчитывать немного реже, например 1 раз в 3–4 последовательных шага, но это не дает значительных преимуществ в скорости вычислений, так как на этапе итерационного вычисления давления возрастают ошибки.

Из-за параллелизма обработки частиц на графическом процессоре трудно заранее сказать, сколько потребуется памяти для хранения массива соседей. Однако за счет несжимаемости жидкости и постоянного размера частицы можно приблизительно оценить максимальное количество соседей, которое должно быть меньше отношения объемов области сглаживания частицы к объему частицы. В нашем случае это отношение приближенно равно 88,7. Но при резком сжатии, особенно при ударах, число соседей может значительно возрасти, поэтому требуется выделить дополнительную память, как минимум для хранения 50–60 адресов соседей. Для снижения задержек при параллельном доступе к памяти GPU доступ к массиву соседних частиц осуществляется таким образом, чтобы каждый поток считывал адрес одного i-го соседа из одного или соседних участков памяти, что достигается за счет расположения данных в памяти как массив, состоящий из max_n массивов из N частиц. Здесь max_n — максимально число соседей, N — число частиц. По адресу, где хранится нулевой сосед, записываем общее количество соседей, затем для доступа к каждому следующему соседу прибавляем к текущему смещению в массиве общее количество частиц.

5.3. Вычисление сил, действующих на частицы

Вычисление сил, действующих на частицы, плотность и давление каждой частицы, выполняется приблизительно одинаково:

1. Инициализация регистра или регистров, в которых будет храниться результат суммирования.
2. Получение количество соседей из массива соседей и запуск цикла по всем соседям.
3. Получение для каждой частицы-соседа позиции и, если требуется, скорости из глобальной памяти. Вычисление расстояния $r = |\mathbf{x}_i - \mathbf{x}_j|$.
4. Вычисление требуемой величины, например, $m_j W(r, h)$ для плотности по формуле (2) и суммирование с результатом.

В случае реализации алгоритма PCISPH на одном из этапов каждой итерации требуется найти максимальную ошибку плотности (рис. 1). Максимальное значение массива при параллельных вычислениях вычисляется с помощью алгоритмов параллельной редукции,

которые вычисляют максимальное значение сначала для небольших блоков, а затем формируют новые блоки из полученных значений. Таким образом, за несколько этапов получим максимальное значение ошибки.

5.4. Обработка граничных условий

В качестве начальных условий моделирования создается массив частиц в виде однородной решетки с расстоянием между частицами, равным 3,008 r, где r — радиус частицы. При таком расстоянии и длине сглаживания $h = 2{,}3r$ для начальных и граничных условий обеспечивается 57 соседей в трехмерном пространстве и 20 в двумерном для каждой частицы и плотность 997,5 кг/м^3. Плотность жидкости ρ_0 = 1000 кг/м^3, но небольшое снижение плотности увеличивает стабильность вычислений за счет небольшой потери точности. Количество соседей 50–60 считается оптимальным для метода SPH в трехмерном пространстве, 20–21 — в двумерном. При большем числе частиц данные эксперимента будут слишком сглажены, при меньшем — часто будет возникать недостаток плотности частиц, что приведет к уменьшению стабильности и некорректному движению частиц. Граничные условия задаются с помощью неподвижных (виртуальных) частиц, которые располагаются в виде решетки размерности на 1 меньше, чем размерность эксперимента с таким же расстоянием между частицами и в 4–5 слоя (рис. 2). Закрашенные круги — частицы жидкости, незакрашенные — неподвижные частицы границ. Область границы заштрихована. Заметим, что центры первого слоя частиц лежат непосредственно на границе. Этим обеспечивается нулевая скорость жидкости на границе.

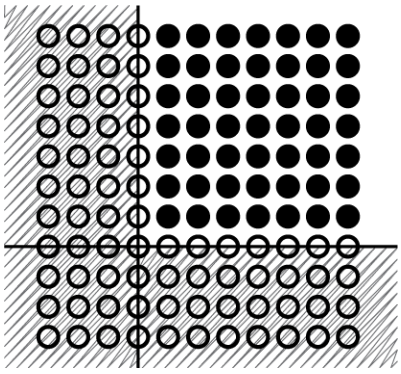

Рис. 2. Начальные и граничные условия, заданные множеством частиц.

Метод SPH можно эффективно реализовать на параллельной архитектуре в том числе за счет того, что все частиц обычно обрабатываются единым образом. Однако в нашем случае

частицы границ оказывают серьезное влияние на производительность в трехмерном случае, так как заполняют большие объемы пространства. Так, при 5-слойной границе и радиуса частицы r = 0,0125 м, в одном из экспериментов требуется более 110000 частиц жидкости и более 150000 частиц границы. Замеры показали, что большую часть времени занимает формирование массива соседей в начале каждого временного шага и подсчет плотности частиц на каждой итерации вычисления силы давления. Так как частицы границ составляют большую часть всех частиц системы, то за счет граничных частиц можно увеличить производительность. Учитывая, что частицы границ не двигаются, а также то, что плотность этих частиц не меняется, если частицы жидкости не воздействуют на них, можно модифицировать схему работы с частицами:

1. На этапе хеширования заполнять также массив, характеризующий частицы – реальные или виртуальные, сенсорные и тому подобные. За счет этого массива на других этапах вычисления будет ясно, какого типа частица обрабатывается данным потоком.
2. На этапе поиска соседей для граничных частиц игнорировать другие граничные частицы. Это позволяет уменьшить число записей в глобальную память на этапе поиска соседей и значительно уменьшить число чтений из глобальной памяти на всех последующих этапах, требующих информации о соседях.
3. На этапе вычисления плотности к плотности граничных частиц добавлять ρ_0.

Таким образом, если виртуальные частицы находятся далеко от частиц жидкости, то их плотность будет равна ρ_0, и эти частицы не будут требовать вычисления силы давления. Но как только частица жидкости приблизится к границе, плотность в данной области возрастет и потребуется вычислить силу давления, которая оттолкнет жидкость от границы. При этом будут задействованы только те граничные частицы, которые входят в число соседей жидкости в данный момент. Остальные граничные частицы не будут требовать вычислений.

Но при таком подходе увеличится радиус воздействия частиц границы. Чтобы не допустить такой ситуации, можно немного отодвинуть граничные частицы от собственно границы, но тогда скорость на границе, возможно, будет ненулевой. Поэтому предлагается в начале моделирования вычислить «реальную» плотность граничных частиц и добавлять к каждой граничной частице не ρ_0, а полученное значение плотности. Следует учитывать, что такой подход будет эффективен только в случае, если границы остаются неизменными на протяжении всего эксперимента.

6. Результаты моделирования

Были реализованы двумерная и трехмерная версии алгоритма PCISPH, результаты моделирования которых представлены в данном разделе.

6.1. Обрушение столба жидкости на плоскости

Дж. Монаган в 1994 г. [2] смоделировал эксперимент из [12] с обрушением столба жидкости с помощью метода SPH, дополненного набором статических виртуальных частиц. Для апробации реализации метода PCISPH на CUDA было проведено моделирование данного эксперимента и произведено сравнение с результатами, полученными ранее. При моделировании использовалась модель горизонтальной поверхности со стенкой высотой 40 м, рядом с которой находится столб жидкости размерами 25x25 м. Также на расстоянии в 50 м от левой стенки находится препятствие в виде прямоугольного треугольника с углом 45^{o} и катетами по 5 м.

В таблице 1 представлены отношения высоты столба жидкости и позиции фронта волны к начальным ширине и высоте столба соответственно. С учетом того, что ошибка измерения времени в оригинальном эксперименте составляла 0.1, сравнение можно считать лишь приблизительным. Значения HT_{exp} и SR_{exp} соответствуют относительным высоте столба и позиции фронта в реальном эксперименте, значения HT_{Mon} и SR_{Mon} — результаты Монагана, HT_{pres} и SR_{pres} — результаты данной работы. Относительное время (t_{rel}), указанное в первой колонке таблицы, переводится в абсолютное (t_{abs}) по формуле $t_{abs} = t_{rel}\sqrt{HT_0/g}$, где HT_0 — высота фронта жидкости в начале эксперимента.

Таблица 1. Сравнение результатов моделирования обрушения столба жидкости на плоскости.

t_{abs}, с (t_{rel})	HT_{exp}	HT_{Mon}	HT_{pres}	SR_{exp}	SR_{Mon}	SR_{pres}
1,134 (0,71)	0,90	0,90	0,89	1,33	1,56	1,49
2,220 (1,39)	0,76	0,75	0,74	2,25	2,50	2,34
3,354 (2,10)	0,57	0,56	0,55	3,22	3,75	3,28
5,111 (3,20)	0,32	0,37	0,35	4,80	5,00	4,76

На рис. 3 представлены результаты моделирования, полученные в моменты времени, указанные в таблице 1. При создании таблицы не учитывались частицы, отлетевшие от основной массы жидкости. Из результатов видно, что реализация алгоритма позволяет моделировать жидкость на плоскости в различных масштабах с высокой точностью.

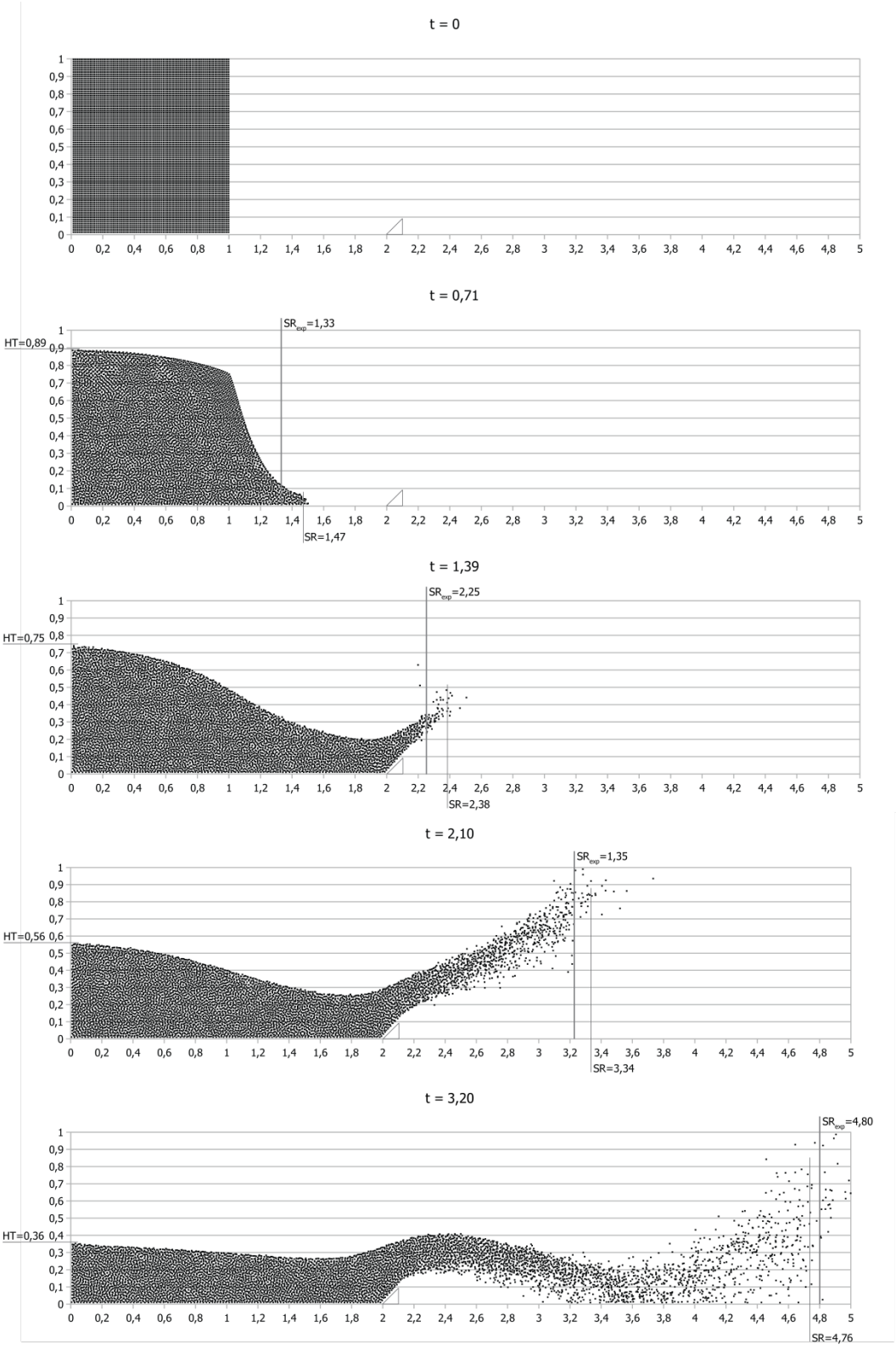

Рис. 3. Результаты моделирования обрушения столба жидкости на плоскости методом PCISPH.

Метод PCISPH был реализован на центральном и графическом процессорах для сравнения производительности и для удобства отладки на CPU. Результаты приведены в таблице 2.

Таблица 2. Сравнение производительности реализаций метода PCISPH на CPU и GPU.

r	N_{fluid}	N_{border}	N	t_{CPU}	t_{GPU}	$t_{\text{CPU}}/t_{\text{GPU}}$	t_{GPU}/N
0,5000	961	887	1848	118	4,9	24	2,652
0,2500	3969	1725	5694	422	5,8	73	1,019
0,2083	5625	2066	7691	566	6,7	84	0,871
0,1500	11025	2872	13897	1083	10,5	103	0,756

В таблице 2 указаны следующие величины:

1) r — радиус частицы;

2) N_{fluid} — количество частиц жидкости, то есть реальных частиц;

3) N_{border} — количество частиц границы, то есть виртуальных частиц;

4) N — общее количество частиц;

5) t_{CPU} — время выполнения одного шага моделирования на центральном процессоре;

6) t_{GPU} — время выполнения одного шага моделирования на графическом процессоре;

7) $t_{\text{CPU}}/t_{\text{GPU}}$ — прирост производительности при использовании GPU;

8) t_{GPU}/N — величина, которая показывает, сколько времени требуется графическому процессору в среднем для обработки одной частицы.

При уменьшении размера частиц увеличивается точность моделирования, но также увеличивается число требуемых итераций для вычисления силы давления. При радиусе частицы 0,15 м в случае ударов время выполнения одного шага моделирования на CPU увеличивается до 1200–1594 мс, а на GPU — до 15–25 мс.

По данным таблицы 2 можно сделать вывод, что реализация алгоритма на GPU становится более эффективной, чем реализация на CPU с увеличением количества частиц.

6.2. Обрушение столба жидкости в пространстве

В Морском Исследовательском Институте Нидерландов (Maritime Research Institute Netherlands, MARIN) был проведен эксперимент обрушения дамбы [13]. Этот эксперимент можно рассматривать как упрощенную модель течения воды по палубе корабля. Вода может

нанести серьезные повреждения кораблю, грузу и персоналу, поэтому важно рассчитывать характер ее движения и силу воздействия воды на груз. В эксперименте 4 датчика H1, H2, H3 и H4 считывают высоту воды, а 8 датчиков P1–P8 считывают давление воды на груз (рис. 4 и 5).

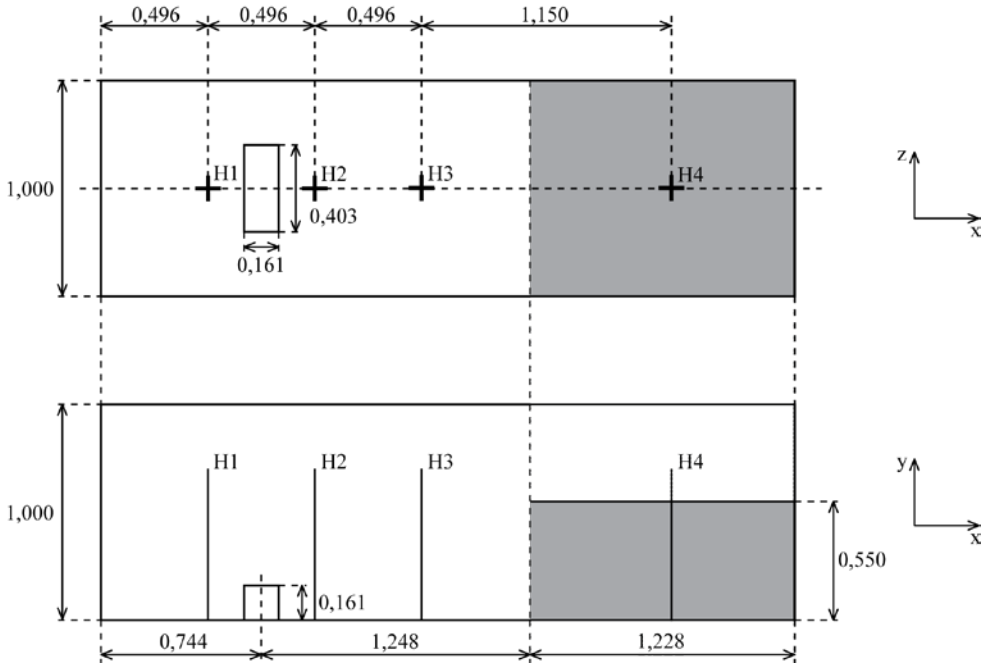

Рис. 4. Схема эксперимента обрушения дамбы: вид сверху и вид сбоку.

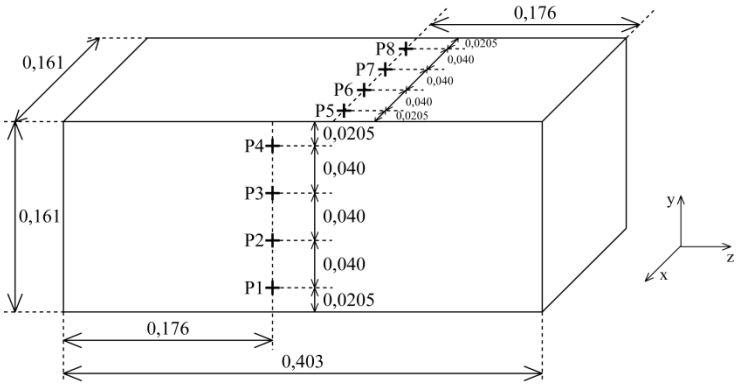

Рис. 5. Схема эксперимента обрушения дамбы: тело и датчики давления.

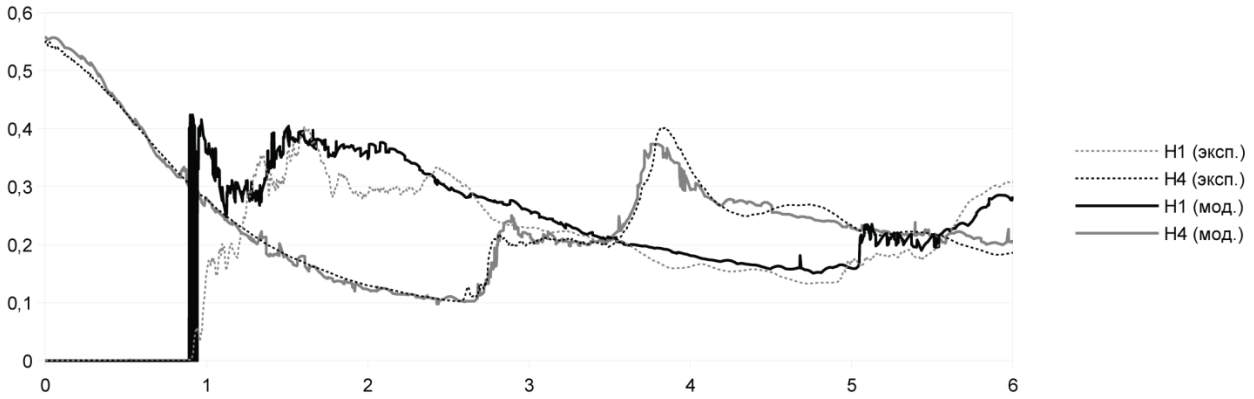

Рис. 6. Сравнение датчиков высоты H1 и H4 в эксперименте и в модели.

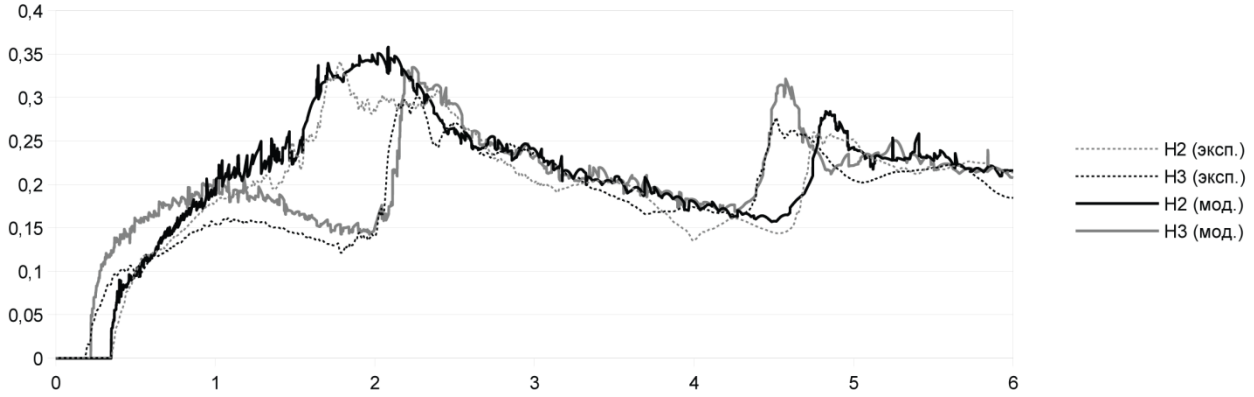

Рис. 7. Сравнение датчиков высоты H2 и H3 в эксперименте и в модели.

На рис. 6 и 7 приведены графики высот столба жидкости в моменты времени от начала эксперимента до 6 с. Экспериментальные данные, полученные в Морском Институте Нидерландов, показаны совместно с данными моделирования. Из графиков видно, что значения, полученные при моделировании, достаточно точно отражают реальный эксперимент. Отклонения возникают лишь при ударах фронта жидкости о препятствия и вызваны появлением брызг жидкости и дискретностью модели частиц.

Для данного эксперимента была измерена производительность при различных параметрах, таких как число частиц, шаг времени. Максимальное число итераций для всех экспериментов равно 18. Моделирование проводилось на системе с графическим адаптером NVIDIA GeForce GTX 470.

Таблица 3. Производительность метода при различных значениях r и Δt.

r, м	N_{fluid}	N_{border}	Δt, с	t_{2c}, с	s	τ, мс
0,022159	16240	48680	0,0015	250,3	124,7	187,0
0,022159	16240	48680	0,001	307,2	153,6	153,6
0,022159	16240	48680	0,0005	363,9	182,0	90,98
0,0125	96512	183388	0,001	1887,9	944,0	944,0
0,0125	96512	144136	0,0005	3594,2	1797,1	898,5

В таблице 3 t_{2c} означает время, затраченное на моделирование первых двух секунд эксперимента, τ означает среднее время вычисления одного шага моделирования, то есть $\tau = t_{2c} / (2\text{ с}) \cdot \Delta t$. Значение s характеризует фактор масштаба по времени, то есть сколько потребуется времени на моделирование одной секунды эксперимента. В нашем случае $s = t_{2c} / (2\text{ с})$.

Видно, что при уменьшении времени шага интегрирования время, затрачиваемое в среднем на один шаг, уменьшается. Это связано с итерационным процессом вычисления силы давления, при котором от ошибки зависит число итераций. При уменьшении Δt уменьшается ошибка, что приводит к пропуску лишних итераций и, следовательно, ускорению расчетов. Однако, число шагов увеличивается быстрее, следовательно, общее время моделирования увеличивается. Также следует учитывать, что если количество итераций становится недостаточным, то точность метода падает, поэтому следует подбирать максимальное количество итераций отдельно для каждого эксперимента. В подтверждение эффективности метода говорит также то, что при увеличении числа частиц в 6 раз (96512/16240 = 5,94) время выполнения при прочих равных параметрах увеличивается также в 6 раз (944,0/153,6 = 6,15), то есть имеет место линейный рост времени вычислений от количества частиц жидкости в модели. Заметим, что количество виртуальных частиц незначительно увеличивает время вычислений, поскольку в множество соседей для виртуальных частиц попадают только частицы жидкости, попавшие в область сглаживания данной виртуальной частицы.

7. Выводы

Метод PCISPH был реализован на центральном и графическом процессорах для сравнения производительности и для удобства отладки. Замеры времени выполнения одного шага

моделирования показали, что с увеличением количества частиц реализация алгоритма на GPU становится более эффективной, чем реализация на CPU. При использовании 13897 частиц реализация на графическом процессоре требует примерно 0,756 мкс для обработки одной частицы, что в 103 раза быстрее, чем однопоточная реализация на центральном процессоре.

Трехмерная модель была реализована лишь на GPU из-за того, что для удовлетворительного с точки зрения точности моделирования жидкости требуется до 10 раз больше частиц, то есть порядка 10^5. При этом центральному процессору требуется более секунды на обработку 10^4 частиц, а с увеличением количества соседей время вычисления увеличится еще примерно в 2,5–3 раза. Графический процессор справляется с такой нагрузкой, и в нашей реализации позволяет моделировать взаимодействие 260000 частиц, из которых 150000 — частицы жидкости и 110000 — частицы границ. Для вычисления одного шага с 16 итерациями требуется до 1,0 с на системе с видеоадаптером GeForce GTX 470 или Tesla C2070.

Представленный алгоритм оптимизирован под большое число статичных граничных частиц, что позволяет снижать вычислительную нагрузку при моделировании больших резервуаров.

Литература

1. Monaghan J. Smoothed Particle Hydrodynamics // Annu. Rev. Astron. Astrophys. 1992. Vol. 30. P. 543–574.

2. Monaghan J. Simulating Free Surface Flows with SPH // Journal of Computational Physics. 1994. Vol. 110. P. 399–406.

3. Блажевич Ю.В., Иванов В.Д., Петров И.Б., Петвиашвили И.В. Моделирование высокоскоростного соударения методом гладких частиц // Матем. моделирование. 1999. Том 11, №1. С. 88–100.

4. Зубов А.Д., Лебедев А.М. Метод сглаженных частиц SPH для расчетов газодинамических задач со сферической и цилиндрической симметриями // Вопросы атомной науки и техники. Серия: Математическое моделирование физических процессов. 2009. Выпуск 1. С. 19–28.

5. Хвостова О.Е., Куркин А.А. Математическое моделирование оползневых цунами методом частиц // Вестник БГТУ им. В.Г. Шухова. 2009. №4. С. 96–100.

6. Liu M., Liu G. Smoothed Particle Hydrodynamics (SPH): an Overview and Recent Developments // Archives of computational methods in engineering. Vol. 17(1). 2010. P. 25--76.
7. Lee E.-S., Violeau D., Issa R., Ploix S. Application of weakly compressible and truly incompressible SPH to 3-D water collapse in waterworks // Journal of Hydraulic Research. 2010. Vol. 48 Extra Issue. P. 50–60.
8. Lee E.-S., Moulinec C., Xu R., Violeau D., Laurence D., Stansby P. Comparisons of weakly compressible and truly incompressible algorithms for the SPH mesh free particle method // Journal of Computational Physics. 2008. V. 227. P. 8417–8436.
9. Solenthaler B., Pajarola R. Predictive-Corrective Incompressible SPH // SIGGRAPH'09: ACM SIGGRAPH 2009 papers. New York, 2009. P. 1–6.
10. Ландау Л.Д., Лифшиц Е.М. Теоретическая физика: в 10 т. Гидродинамика. М.: Физматлит, 2006. T.VI. 736 с.
11. Becker M., Teschner M. Weakly compressible SPH for free surface flows Eurographics //ACM SIGGRAPH Symposium on Computer Animation 2007, pp. 1–8.
12. Martin J.C., Moyce W.J. Part IV. Collapse of Free Liquid Columns on a Rigid Horizontal Plane // Philosophical Transactions of the Royal Society of London. 1952. Vol. 244. P. 312–324.
13. Issa R., Violeau D. Test-case 2, 3d dambreaking, Release 1.1. // ERCOFTAC, SPH European Research Interest Community SIG, Electricit`e De France, Laboratoire National d'hydaulique et Environnement. 2006. [Электронный ресурс]. URL: http://wiki.manchester.ac.uk/spheric/index.php/Test2 (29.03.2012).

SCIENTIFIC PERIODICAL OF THE BAUMAN MSTU
SCIENCE and EDUCATION
EL № FS 77 - 48211. №0421200025. ISSN 1994-0408
electronic scientific and technical journal

Incompressible fluid simulation on CUDA using SPH method

 # 07, July 2012
DOI: 10.7463/0712.0423582

Suravikin A.Yu.

Russia, Omsk F.M. Dostoevsky State University
tizona_del_cid@rambler.ru

In the article the author describes the Smoothed Particle Hydrodynamics (SPH) method theory and its parallel implementation for simulation of incompressible fluids. Implementation of Predictive-Corrective Incompressible SPH (PCISPH) method is described in detail. Parallel techniques are introduced to specify and to process initial and boundary conditions in fluid simulation. The author presents results for two- and three-dimensional cases of collapse of column of fluid such as comparison of simulation with real experiments and with results obtained by other authors, performance comparison of CPU and GPU implementations.

Publications with keywords: parallelism, graphics processors, SPH, CUDA, incompressible fluids, Smoothed Particle Hydrodynamics, fluid simulation
Publications with words: parallelism, graphics processors, SPH, CUDA, incompressible fluids, Smoothed Particle Hydrodynamics, fluid simulation

References

1. Monaghan J. Smoothed Particle Hydrodynamics. *Annual Review of Astronomy and Astrophysics*, 1992, vol. 30, pp. 543–574.

2. Monaghan J.J. Simulating free surface flows with SPH. *Journal of Computational Physics*, 1994, vol. 110, no. 2, pp. 399–406. DOI: 10.1006/jcph.1994.1034.

3. Blazhevich Iu.V., Ivanov V.D., Petrov I.B., Petviashvili I.V. Modelirovanie vysokoskorostnogo soudareniia metodom gladkikh chastits [High impact modeling with smooth particle]. *Matematicheskoe modelirovanie*, 1999, vol. 11, no. 1, pp. 88-100.

4. Zubov A.D., Lebedev A.M. Metod sglazhennykh chastits SPH dlia raschetov gazodinamicheskikh zadach so sfericheskoi i tsilindricheskoi simmetriiami [The method of SPH smoothed particles for

calculations of gas-dynamic problems with spherical and cylindrical symmetry]. *Voprosy atomnoi nauki i tekhniki. Ser. Matematicheskoe modelirovanie fizicheskikh protsessov* [Problems of Atomic Science and Engineering. Ser. Mathematical Modeling of Physical Processes], 2009, no. 1, pp. 19-28.

5. Khvostova O.E., Kurkin A.A. Matematicheskoe modelirovanie opolznevykh tsunami metodom chastits [Mathematical modeling of landslide tsunami by particle method]. *Vestnik BGTU im. V.G. Shukhova* [Herald of the Shukhov BSTU], 2009, no. 4, pp. 96-100.

6. Liu M., Liu G. Smoothed particle hydrodynamics (SPH): An overview and recent developments. *Archives of Computational Methods in Engineering*, 2010, vol. 17, no. 1, pp. 25-76.

7. Lee E.-S., Violeau D., Issa R., Ploix S. Application of weakly compressible and truly incompressible SPH to 3-D water collapse in waterworks. *Journal of Hydraulic Research*, 2010, vol. 48, suppl. 1, pp. 50–60. DOI: 10.1080/00221686.2010.

8. Lee E.-S., Moulinec C., Xu R., Violeau D., Laurence D., Stansby P. Comparisons of weakly compressible and truly incompressible algorithms for the SPH mesh free particle method. *Journal of Computational Physics*, 2008, vol. 227, no. 18, pp. 8417–8436. DOI: 10.1016/j.jcp.2008.06.005

9. Solenthaler B., Pajarola R. Predictive-corrective incompressible SPH. *ACM Transactions on Graphics*, 2009, vol. 28, no. 3, art. no. 40. DOI: 10.1145/1531326.1531346.

10. Landau L.D., Lifshits E.M. *Teoreticheskaia fizika. T. 6. Gidrodinamika* [Theoretical physics. Vol. 6. Hydrodynamics]. Moscow, Fizmatlit Publ., 2006. 736 p.

11. Becker M., Teschner M. Weakly compressible SPH for free surface flows. *Proc. 2007 ACM SIGGRAPH/Eurographics Symp. on Computer Animation (SCA'07)*. 2007, pp. 209-217.

12. Martin J.C., Moyce W.J. Part IV. Collapse of Free Liquid Columns on a Rigid Horizontal Plane. *Philosophical Transactions of the Royal Society of London*, 1952, vol. 244, pp. 312–324.

13. Issa R., Violeau D. *Test-case 2. 3D dambreaking, Release 1.1*. ERCOFTAC, SPH European Research Interest Community SIG, Electricit`e De France, Laboratoire National d'hydaulique et Environnement. 2006. Available at: http://wiki.manchester.ac.uk/spheric/index.php/Test2. Accessed March 29, 2012.