Моделирование жидкости с использованием системы частиц

Р.В. Реутов, Т.А. Приходько

***Аннотация* – В статье рассматривается одна из важных проблем современной компьютерной графики – проблема физически-достоверной анимации таких сложных объектов как вода, дым, огонь. Эти объекты могут принимать сложные физические формы, обтекать другие объекты на сцене. Их поведение зависит от множества факторов: воздействие гравитации; потоков других газов и жидкостей; давления. Для описания этого динамического поведения используются вычислительно-сложные уравнения, выполнение которых может быть слишком затратным для области компьютерной графики, особенно, если необходимо создавать анимацию в реальном времени. Без достоверной физической модели воды и дыма крайне трудно создать фотореалистичную анимацию, так как неверное поведение описанных объектов сразу бросается в глаза.**

В данной статье рассматривается физическая модель воды, построенная на основе неявной системы частиц. Алгоритм является гибридным, и использует как систему частиц, так и векторные поля, для моделирования поведения воды и базируется на существующем алгоритме PIC/FLIP.

Рассматривается подход к реализации алгоритма используя технологию Cuda – программно-аппаратное решение GPGPU, хорошо подходящее для алгоритмов семейств PIC и FLIP. Однако, подход к решению задачи может быть использован и на других программных решениях, таких как OpenMP.

***Ключевые слова* – система N-тел, компьютерное моделирование, компьютерная графика, гибридный алгоритм, частицы Лапласа, GPGPU.**

I. ПРЕДСТАВЛЕНИЕ ЖИДКОСТИ

Существует два способа представления жидкостей во время моделирования. Представления основаны на совершенно разных точках зрения на систему – одна рассматривает жидкость как совокупность множества частиц, называемых атомами, а вторая рассматривает пространство, разбитое на ячейки, через которые течет поток жидкости.

Представление жидкости на основе частиц называют «Лагранжевым представлением». Представление на основе частиц породило большое количество моделей жидкости, самым распространенным из которых является алгоритм SPH (гидродинамика сглаженных частиц). Недостатком данных алгоритмов является то, что расчеты в регионах с малой плотностью частиц становятся весьма неточными, в связи с недостатком информации, хранящейся в ближайших частицах.

Иной подход к представлению жидкости, основывается на определенных местах в пространстве, через которые течет жидкость с течением времени. Пространство разбивается на множество ячеек, образующих сетку, после чего в каждой из ячеек вычисляется имеющийся поток воды. Такое представление позволяет избегать проблем, связанных с низкой плотностью жидкости в области, однако возникают иные проблемы. Чтобы поверхность воды была достаточно детализированной, необходимо, чтобы сетка состояла из большого количества ячеек. Методы, основанные исключительно на сетке, должны удовлетворять критерию Найквиста – «частота дискретизации должна быть в два раза больше, чем у самой высокой составляющей».

Возможно использовать, в том числе, гибридный подход, в котором модель использует как частицы, так и поля – используется Лагранжево представление для захвата поверхности жидкости, а вспомогательная сетка для обеспечения точного сохранения массы.

При моделировании жидкости, с точки зрения физики, возможно использование двух разных подходов для ее математического описания. Первым подходом являются уравнение Навье-Стокса – система дифференциальных уравнений в частных производных, которые описывают движения вязкой Ньютоновской жидкости [1]. Система для несжимаемой жидкости состоит из двух уравнений: уравнение неразрывности и уравнения движения. Формула 1 описывает уравнения Навье-Стокса в векторном виде:

$$\frac{\partial \vec{u}}{\partial t} = -(u \cdot \nabla)\vec{u} + \nu \Delta \vec{u} - \frac{1}{\rho}\nabla p + \vec{f} \qquad (1)$$

где Δ – векторный оператор Лапласа ∇ – оператор Набла, ρ – плотность, p – давление, ν – коэффициент кинематической плотности, $\vec{u}$ – векторное поле скорости, $\vec{f}$ – векторное поле массовых сил.

Вторым подходом моделирования является использование уравнение Пуассона, вычисление которого является более простой задачей. Однако, необходимо учитывать, что уравнение Пуассона описывает некоторое поле, которое не обязательно

является представлением жидкостной системы. Зачастую, уравнение использование для представления электростатических и гравитационных полей. В декартовой системе координат уравнение Пуассона принимает вид 2 [2].

$$(\frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2}+\frac{\partial^2}{\partial z^2})\varphi(x,y,z)=f(x,y,z)$$
$$\nabla^2\varphi=f \qquad (2)$$

где f – некоторая вещественная или комплексная функция, а φ – некоторая искомая функция.

В дальнейшем формула 2 будет использоваться в вычислениях поля масс, так как данное представление является наиболее удобным и простым в вычислительно простым.

II. ОПЕРАТОРЫ ВЕКТОРНЫХ ПОЛЕЙ

Так как в рассматриваемом гибридном методе используются векторные поля, необходимо определить операторы для работы с векторным полем, чтобы понимать, как изменяется поле скоростей и как сохраняется масса жидкости при работе с полем.

Формула 3 показывает оператор градиента, работающий со скалярным векторным полем и производит векторное поле, элементы векторов которого являются соответствующими частными производными для компонентов исходного поля. Результаты применения оператора следует интерпретировать как локальное изменения скалярного поля q.

$$\nabla q=\left(\frac{\partial q}{\partial x},\frac{\partial q}{\partial y},\frac{\partial q}{\partial z}\right) \qquad (3)$$

Формула 4 показывает оператор дивергенции. Оператор дивергенции связан с оператором градиента и включает в себя те же частичные производные, но отражает векторное поле на скалярное. Оператор определяет насколько расходятся входящий и исходящий из точки поток.

$$\nabla\cdot u=\frac{\partial u}{\partial x}+\frac{\partial u}{\partial y}+\frac{\partial u}{\partial z} \qquad (4)$$

Наиболее комплексным оператором является оператор Лапласа или Лапласиан, представленный формулой 5. Оператор работает над скалярным полем q. Оператор эквивалентен последовательному взятию функции градиента и дивергенции [3].

$$\nabla^2 q=\frac{\partial^2 q}{\partial x^2}+\frac{\partial^2 q}{\partial y^2}+\frac{\partial^2 q}{\partial z^2} \qquad (5)$$

III. СТУПЕНЧАТАЯ СЕТКА

Необходима возможность применять непрерывные операторы к дискретной сеточной структуре. Для этого необходимо дифференцировать операторы. Для этого необходимо ввести специальный вид сетки – шахматную сетку маркеров и ячеек. Принципиальное отличие такой сетки от обычной состоит в том, что компоненты поля скоростей разделены и совмещены, а точки выборки поля скорости совпадают с границами граней ячейки. Заметим, что все скаляры, переносимые полем скоростей, имеют точки отчета, расположенные в центре ячейки. Структура сетки представлена на рисунке 1.

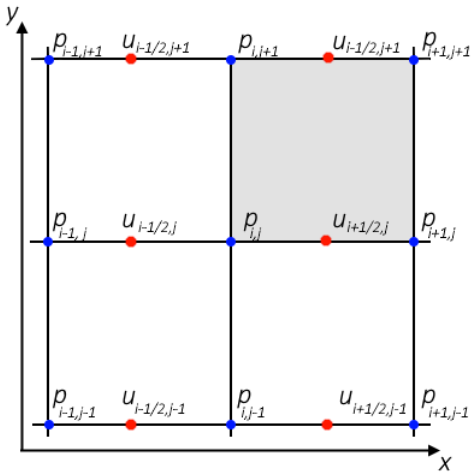

Рис. 1 Структура шахматной сетки

Такая структура имеет серьезное преимущество – вычисление дискретных эквивалентов ранее определенных операторов значительно упрощается. Структура вводит полуиндекс, в котором положительная сторона полуиндекса – это значение выборки, хранящейся на грани используемой ячейкой совместно с последующей.

В литературе такая структура известна как «marker-and-cell (MAC) velocity field» [3]. Она используется во множестве практически применяемых алгоритмов.

В уравнении 6 рассматривается дискретный градиент, где s – размер ячейки. Градиент применяется к скаляру q, который хранится в центре ячеек. Таким образом, оценочная позиция должна находиться между двумя соседними ячейками.

$$(\nabla q)_{i+\frac{1}{2},j,k} = \frac{q_{i+1,j,k} - q_{i,j,k}}{s}$$
$$(\nabla q)_{i,j+\frac{1}{2},k} = \frac{q_{i,j+1,k} - q_{i,j,k}}{s} \quad (6)$$
$$(\nabla q)_{i,j,k+\frac{1}{2}} = \frac{q_{i+1,j,k+1} - q_{i,j,k}}{s}$$

Определение оператора дискретной дивергенции показано в уравнении 7. Так как дивергенция работает с парами компонентов скоростей, ее расчетная позиция должна быть в центре ячейки.

$$(\nabla \cdot \bar{u})_{i,j,k} = \frac{u_{i+1/2,j,k} - u_{i-1/2,j,k}}{s} + \frac{u_{i,j+1/2,k} - u_{i,j-1/2,k}}{s} + \frac{u_{i,j,k+1/2} - u_{i,j,k-1/2}}{s} \quad (7)$$

При комбинации операторов 4 и 5 будет получен оператор Лапласа, показанный в формуле 8.

$$(\nabla^2 q)_{i,j,k} = \frac{q_{i+1,j,k} + q_{i-1,j,k}}{s^2} + \frac{q_{i,j+1,k} + q_{i,j-1,k}}{s^2} + \frac{q_{i,j,k+1} + q_{i,j,k-1}}{s^2} - \frac{6q_{i,j,k}}{s^2} \quad (8)$$

Заметим, что оператор Лапласа и оператор Дивергенции определены в центре ячейки. Уравнения Пауссона, используемые в физике для описания поведения жидкостей, имеют форму, показанную в уравнении 9.

$$\nabla \cdot \bar{u} = \nabla^2 q \quad (9)$$

На основании подобных вычислений в 1999 году был предложен метод Stable Fluids, однако сейчас данный метод является непопулярным, в связи с низкой точности решения, относительно его вычислительной сложности. [5]

IV. ГИБРИДНЫЙ МЕТОД МОДЕЛИРОВАНИЯ

Как было описано ранее, каждая из двух точек зрения на моделируемую жидкость имеет свои достоинства и недостатки. Имеет смысл воспользоваться как Лагранжевым методом частиц, так и методом Эйлеровой сетки одновременно – позволить Лагранжевым частицам управлять адвективной частью моделирования, а Эйлеровой сетки обрабатывать ограничения давления и несжимаемости.

Алгоритм использует воксельное представление сцены и частиц. Воксельное представление позволяет легко и эффективно представить трехмерную сетку [6].

Существуют методы моделирования PIC и FLIP, представляющие оба описанных подхода. Отдельно методы имеют некоторые недостатки – жидкости, моделируемые методом FLIP получаются достаточно вязкими [7], а PIC методом – имеют поверхностные шумы [8]. Рассмотрим алгоритм гибридного алгоритма, использующего оба метода одновременно. На рисунке 2 представлена принципиальная блок-схема алгоритма.

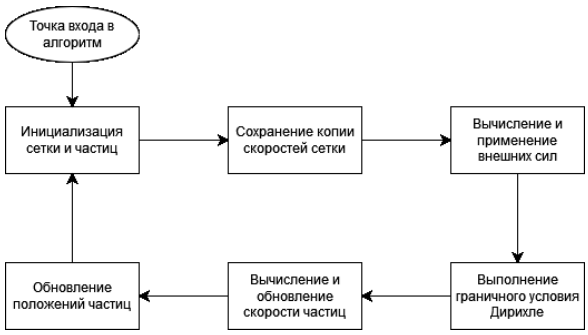

Рис. 2 Блок-схема алгоритма

Каждый воксель, классифицируемый как текучий, будет иметь засеянные внутрь частицы жидкости. С помощью трилинейной интерполяции обновляются скорости частиц в сетке.

Граничное условие Дирихле утверждает, что не должно быть ни втекания, ни вытекания сплошных ячеек, когда выполняется условие 10, где n-нормаль к соседней ячейки

$$Vn = 0 \quad (10)$$

То есть, если жидкая ячейка имеет соседнюю твердую ячейку, на которую указывает компонент скорости, то необходимо обнулить компонент скорости, указывающий на эту плотную ячейку. Так как вычисления выполняются на сетке, добиться этого достаточно просто.

Так как поле скорости, сохраняющее массу, является синонимом бездивергентного поля скорости, сохранение массы жидкости равносильно обеспечению нулевой дивергенции на поле скоростей, переносящим массу. Это самый сложный шаг моделирования в вычислительном плане, поскольку для вычисления этого шага требуется вычисление уравнения Пауссона. Так как оператор Лапласа определен как сумма частных производных второго порядка, можно свести решение уравнение Пауссона к массивной системе линейных уравнений.

Разложение Гельмгольца-Ходжа утверждает, что любое векторное поле V можно представить в виде суммы бездивергентной части V_{df} и свободной части V_{cf}. Кроме того, градиент любого скалярного поля, по определению, не имеет завихрений, что позволяет вычислить уравнение 11.

$$V = V_{df} + V_{cf}$$
$$V = V_{df} + \nabla q$$
$$\nabla V = \nabla V_{df} + \nabla \cdot \nabla q \quad (11)$$

Полученное уравнение 11 может быть упрощено, так как V_{df} является бездевиргентной частью, результатом применения оператора дивергенции на нее будет нуль. Также, исходя из данного определения оператора Лапласа в формуле 3,

появляться возможность упростить уравнение 11, приведя его к виду 12.

$$\nabla V = \nabla^2 q \qquad (12)$$

Уравнение 12 является уравнение Пауссона котором q является неизвестным скалярным полем, которое является решением уравнения. Данное уравнение необходимо решить для нахождения значений скалярного поля и распределения масс.

V. ПОДХОД К РЕАЛИЗАЦИИ ПОСРЕДСТВАМ CUDA

Алгоритм процесса моделирования может быть эффективно разбит на множество потоков с несколькими точками синхронизации. Поэтому предлагается использовать программно-архитектурный комплекс Cuda для решения данной задачи. Однако, описанные ниже сведения применимы, в том числе, к реализации параллельных вычислений посредствам иных платформ – как традиционного параллельного вычисления на CPU, так и других GPUPU платформ, таких как OpenCL. С его помощью возможно произвести моделирование посредствам ресурсов видеокарты – GPU. Реализация параллельного алгоритма потребует использование особого подхода к хранению переменных и индексов, что ее усложняет, относительно однопоточной реализации алгоритма [9].

Процесс моделирования может быть представлен как цикл, состоящий из вызова 4 слабосвязанных процедур [10], внутренняя реализация каждая из которых может быть выполнена параллельно. На рисунке 3 представлена схема главного цикла моделирования, в которой размечены четыре основные процедуры моделирования и переменные, используемые во время выполнения процедур. Каждая процедура является отдельной Cuda-программой со своими входными данными. Такие программы называются ядрами (Kernel).

Описанная может быть выполнена асинхронно с другими процедурами и модулями программы, таким образом ускоряя процесс моделирования, при копировании результатов фрейма. Такой подход потребует копирования от одной до двух групп переменных, в зависимости от выполняемой асинхронной задачи: Particle Variables и/или Field Variables.

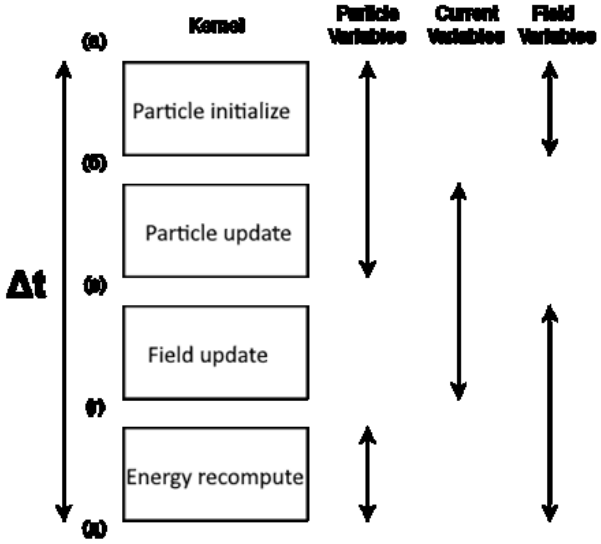

Рис. 3 основной цикл моделирования совмещенный с диаграммой блокировки переменных

При помощи Δt обозначим время между проходящее между двумя фреймами моделирования; как F обозначим внешние силы, действующие на модель, такие как гравитация или взаимодействие с иными объектами. Тогда поле V каждый фрейм изменяется по формуле 13.

$$V_{new} = V_{old} + F\Delta t \qquad (13)$$

Таким образом, задача программы сводится к построению некоторую последовательность фреймов моделирования $T = (t_0 ... t_n)$, где $t_{m+1} = t_m + \Delta t$. Использование дискретного времени повлечет за собой дополнительное внедрение методов интерполяции в алгоритм.

Кроме того, имеется возможность произвести параллельные вычисления шага инициализации модели: начальное распределение частиц и их масс в поле. Парализация возможна путем разбивания множества частиц на отдельные непересекающиеся подмножества. В данной статье параллельная реализация данного шага не рассматривается.

Предлагается использовать три множества переменных – переменные, представляющие описания системы частиц (PIC подход), переменные представляющие описание поля (FLIP подход) и множество переменных, описывающих текущие промежуточные вычисления, которые еще не применены к сохраненным данным модели, так как не синхронизированы с ее второй частью.

Процедура Particle initialize производит начальную инициализацию положения частиц и поля для фрейма моделирования, одновременно с этим сохраняя копии скоростей сетки. Копии необходимы для решения проблемы гонки данных во время выполнения ядра.

Перед выполнением данной процедуры возможно начинать выполнение дополнительных асинхронных задач, например, рендеринга кадра анимации, путем копирования данных частиц предыдущего фрейма.

Действия копирования и передачи данных могут быть встроены в точку (а).

Процедура Particle update вычисляет текущие запланированные изменения положения частиц на сетке. При чем, изменения не применяются на данном этапе, так как на текущем шаге нет гарантии сохранения массы. На рисунке 4 изображена схема перемещения частиц.

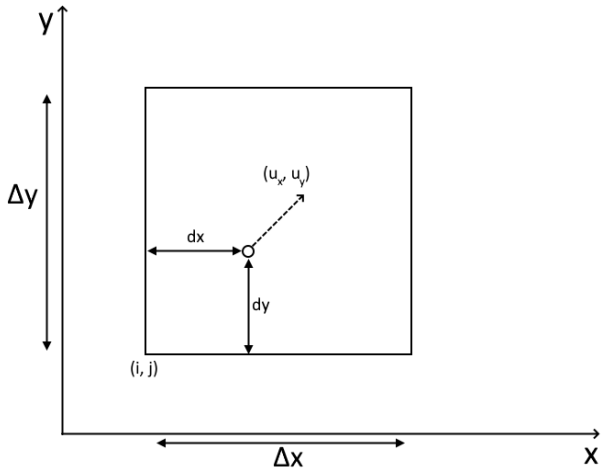

Рис. 4 схема запланированного перемещения одной из частиц

Процедура Field Update производит обновление поля частиц исходя из запланированных изменений частиц на прошлом шаге, производит корректировку их положения, согласно данным FLIP части алгоритма. После чего производится обновления энергий и массы в поле, синхронизация данных в поля и системы частиц.

На данном этапе очень важным решением является выбор метода интерполяции данных. Метод интерполяции может сильно повлиять на результаты моделирования, что позволяет использовать метод интерполяции как один из параметров модели при ее реализации. В зависимости от того, какие требования предъявляются модели, требуются разные методы интерполяции. Кроме того, метод интерполяции может повлиять на время просчета кадра моделирования. На требования к модели может влиять, то, где модель будет применяться – например, мультипликация и инженерные вычисления предъявляют разные требования к итоговому результату.

Финальная процедура Energy Recompute является наиболее затратной с вычислительной точки зрения, так как в ней требуется вычислить решение уравнение Пуассона 9. Данный шаг является одним из самых сложных в вычислительном плане, так как для его решения требуется применение численных методов решения систем дифференциальных уравнений. В зависимости от выбора метода решения системы уравнений вычислительная сложность шага может изменяться.

При вычислении системы уравнений рекомендуется соблюдать баланс точности вычислений и их сложности. Например, для задач, связанных с компьютерной графикой, достаточно относительно небольшой точности вычислений.

Исходя из схемы цикла 3, можно заметить, что возможно выполнение алгоритма с минимальным количеством ожиданий. Синхронизации потоков необходимы в критических точках, отмеченных как (а), (б), (в), (г). Точка (д) совпадает с точкой (а), начиная со второго фрейма моделирования.

Так как переменные поля и ячеек представляют собой некоторую ячеистую структуру, они могут быть упакованы как многомерные или одномерные массивы, что позволяет легко вычислять новые значения посредствам множества ядер. Имеется возможность представить вычисления как некоторое множество операций над векторами и матрицами, что дополнительно облегчает выполнение алгоритма посредствам GPU.

VI. РЕЗУЛЬТАТЫ РАБОТЫ

В результате работы были рассмотрены способы моделирования жидкости посредствам частиц и Эйлеровых полей и предложен алгоритм, совмещающий оба подхода к моделированию жидкости.

Разработан алгоритм и обобщенный подход к его реализации посредствам GPUPU при помощи таких инструментов, как Cuda и OpenCL, который был рассмотрен на примере реализации при помощи программно-аппаратного комплекса Cuda.

Предложенный подход к реализации модели жидкости может быть использован не только для моделирования воды, но и других жидкостей, чего можно добиться, путем изменения параметров алгоритмов FLIP и PIC, изменяя такие параметры жидкости как вязкость и масса, что позволяет моделировать и иные жидкости.

Ускорение алгоритма посредствам обработки модели посредствам GPU, а конкретно при помощи Cuda, позволяет значительно ускорить процесс обработки модели. Появляется возможность обработки моделей в реальном времени.

В дальнейшем планируется развитие проекта и дальнейшее исследование моделей жидкостей и газов, основанных на системах частиц. Запланировано исследование способов и подходов к визуализации моделирования в реальном времени посредствам алгоритмов, основанных на SDF-функциях и алгоритмов построения полигональных мешей на основе системы частиц. Отдельного внимания заслуживают алгоритмы, основанные на исключительно воксельном представлении жидкости. Воксельное представление может быть использовано как аналог MAC-сетки, которая сможет, в том числе, обеспечить и сохранение массы и инерции частиц, без использования дополнительных структур.

Библиография

[1] Acheson, D. J. 1990. Elementary Fluid Dynamics // Oxford Applied Mathematics and Computing Science Series, Oxford University Press – ISBN 978-0-19-859679-0 – Текст: непосредственный
[2] Robert Bridson. 2015. Fluid Simulation for Computer Graphics // MDPI AG, Sensors — ISBN 978-1-56881-326-4. — Текст : непосредственный.
[3] Laplace operator. 1994. Encyclopedia of Mathematics // EMS Press // Текст: электронный – URL: https://encyclopediaofmath.org/index.php?title=Laplace_operator (дата обращения: 28.06.2023)
[4] Fluid Flow for the Rest of Us: Tutorial of the Marker and Cell Method in Computer Graphics. 2005. Текст: электронный. — URL: https://cg.informatik.uni-freiburg.de/intern/seminar/gridFluids_fluid_flow_for_the_rest_of_us.pdf (дата обращения: 28.06.2023)
[5] Zou, C., Yin, Y.Multi-phase Fluid Simulation Based on Narrow-Band FLIP Method. // Текст: электронный. 2018. https://doi.org/10.1007/s13319-018-0173-z (дата обращения: 03.07.2023)
[6] Chmielewski, Sz., Tompalski, P. 2017 Estimating outdoor advertising media visibility with voxel-based approach. Applied Geography. // Текст: электронный // Социальная сеть для учёных: официальный сайт // URL: https://www.researchgate.net/publication/318744436_Estimating_outdoor_advertising_media_visibility_with_voxel-based_approach (дата обращения: 15.07.2023)
[7] Overview of Indoor Navigation Techniques. 1988. — Текст : электронный // Социальная сеть для учёных: официальный сайт. — URL: https://www.researchgate.net/publication/222452290_FLIP_A_Low-Dissipation_Particle-in-Cell_Method_for_Fluid_Flow (дата обращения: 10.07.2023).
[8] Birdsall, Charles K.; A. Bruce Langdon (1985). Plasma Physics via Computer Simulation. McGraw-Hill. ISBN 0-07-005371-5. – Текст: непосредственный.
[9] CUDA C++ Programming Guide. 2023 – Текст: электронный – URL: https://docs.nvidia.com/cuda/pdf/CUDA_C_Programming_Guide.pdf
[10] Feynman, Richard P. 1982. Simulating physics with computers // International Journal of Theoretical Physics. 21 (6–7): 467–488 // текст: электронный // URL: https://web.archive.org/web/20170812065758/http://www.mrtc.mdh.se/~gdc/work/ARTICLES/2014/3-CiE-journal/Background/SimulatingPhysicsWithComputers.pdf (дата обращения: 01.07.2023)

Сведения об авторах:

Приходько Татьяна Александровна, кандидат технических наук, доцент кафедры вычислительных технологий факультета компьютерных технологий и прикладной математики.
Кубанский государственный университет г. Краснодар, Россия (350040. Россия, г. Краснодар, ул. Ставропольская, д.149.)
pr.tatyana@gmail.com
(+7)967-314-55-68
ORCID: https://orcid.org/0000-0001-5137-2064

Реутов Роман Владимирович, магистрант кафедры вычислительных технологий факультета компьютерных технологий и прикладной математики, Кубанский государственный университет г. Краснодар, Россия
(350040. Россия, г. Краснодар, ул. Ставропольская, д.149.) ORCID: ORCHID: 0009-0001-8131-5623 peytob23@gmail.com

Статья получена: 22.10.2023

Все авторы прочитали и одобрили окончательный вариант рукописи.

Modeling a fluid using a particle system

R.V. Reutov, T.A. Prikhodko

Abstract - **The article discusses one of the important problems of modern computer graphics - the problem of physically accurate animation of such complex objects as water, smoke, fire. These objects can take on complex physical shapes and flow around other objects on the stage. Their behavior depends on many factors: the influence of gravity; flows of other gases and liquids; pressure. To describe this dynamic behavior, computationally complex equations are used, which can be prohibitively expensive for the computer graphics field to execute, especially if the animation needs to be done in real time. Without a reliable physical model of water and smoke, it is extremely difficult to create photorealistic animation, since the incorrect behavior of the described objects immediately catches the eye.**

This article discusses a physical model of water built on the basis of an implicit particle system. The algorithm is a hybrid one, using both a particle system and vector fields to model the behavior of water and is based on the existing PIC/FLIP algorithm.

An approach to implementing the algorithm using Cuda technology is considered - a GPGPU hardware and software solution that is well suited for algorithms of the PIC and FLIP families. However, the approach to solving the problem can be used with other software solutions, such as OpenMP.

Keywords – N-body system, computer modeling, computer graphics, hybrid algorithm, Laplace particles, GPGPU.

REFERENCES

[1] Acheson, D. J. 1990. Elementary Fluid Dynamics // Oxford Applied Mathematics and Computing Science Series, Oxford University Press – ISBN 978-0-19-859679-0

[2] Robert Bridson. 2015. Fluid Simulation for Computer Graphics // MDPI AG, Sensors — ISBN 978-1-56881-326-4.

[3] Laplace operator. 1994. Encyclopedia of Mathematics // EMS Press // Text: electronic – URL: https://encyclopediaofmath.org/index.php?title=Laplace_operator (Retrieved: 28.06.2023)

[4] Fluid Flow for the Rest of Us: Tutorial of the Marker and Cell Method in Computer Graphics. 2005. Text: electronic. — URL: https://cg.informatik.uni-freiburg.de/intern/seminar/gridFluids_fluid_flow_for_the_rest_of_us.pdf (дата обращения: 28.06.2023)

[5] Zou, C., Yin, Y.Multi-phase Fluid Simulation Based on Narrow-Band FLIP Method. // Text: electronic. 2018. https://doi.org/10.1007/s13319-018-0173-z (Retrieved: 03.07.2023)

[6] Chmielewski, Sz., Tompalski, P. 2017 Estimating outdoor advertising media visibility with voxel-based approach. Applied Geography. // Text: electronic // Social Network for Scientists: official website // URL: https://www.researchgate.net/publication/318744436_Estimating_outdoor_advertising_media_visibility_with_voxel-based_approach (Retrieved: 15.07.2023)

[7] Overview of Indoor Navigation Techniques. 1988. — Text: electronic // Social Network for Scientists: official website. — URL: https://www.researchgate.net/publication/222452290_FLIP_A_Low-Dissipation_Particle-in-Cell_Method_for_Fluid_Flow (Retrieved: 10.07.2023).

[8] Birdsall, Charles K.; A. Bruce Langdon (1985). Plasma Physics via Computer Simulation. McGraw-Hill. ISBN 0-07-005371-5.

[9] CUDA C++ Programming Guide. 2023 – Text: electronic – URL: https://docs.nvidia.com/cuda/pdf/CUDA_C_Programming_Guide.pdf

[10] Feynman, Richard P. 1982. Simulating physics with computers // International Journal of Theoretical Physics. 21 (6–7): 467–488 // Text: electronic // URL: https://web.archive.org/web/20170812065758/http://www.mrtc.mdh.se/~gdc/work/ARTICLES/2014/3-CiE-journal/Background/SimulatingPhysicsWithComputers.pdf (Retrieved: 01.07.2023)

Information about the Authors:

Tatyana A. Prikhodko, Candidate of Science (Techniques), associate Professor of the Department of computing technologies, Faculty of Computer technologies and Applied Mathematics.
Kuban state University Krasnodar, Russia
(350040. 149, Stavropol str., Krasnodar, Russia.)
pr.tatyana@gmail.com (+7)967-314-55-68
ORCID: https://orcid.org/0000-0001-5137-2064

Roman V. Reutov, undergraduate student of the Department of computing technologies, Faculty of Computer technologies and Applied Mathematics, Kuban state University Krasnodar, Russia
(350040. 149, Stavropol str., Krasnodar, Russia.)
ORCHID: https://orcid.org/0009-0001-8131-5623
peytob23@gmail.com

All authors read and approved the final version manuscripts.