Московский государственный технический
университет имени Н. Э. Баумана

Факультет «Информатика и системы управления»
Кафедра «Программное обеспечение ЭВМ
и информационные технологии»

Расчётно-пояснительная записка
к курсовому проекту по компьютерной графике на тему

Моделирование поведения жидкости в системе из шара и внешнего контейнера методом гидродинамики сглаженных частиц

Исполнитель: студент ИУ7-51 Павелко П. Ю. _____

Руководитель: к. т. н., доцент ИУ7 Филиппов М. В. _____

Москва, 2015

Содержание

Введение

Задача моделирования поведения жидкости находит много применений в различных отраслях. В физике, технике и математике основные требования предъявляются к физической корректности и точности моделирования, а не к визуальному результату, причём основная задача состоит именно в минимизации погрешности результата. В компьютерных играх и кинематографе наоборот, основополагающим является визуальный результат, а физическая корректность может быть частично нарушена.

Моделирование гидродинамики применяется во многих отраслях:

— самолетостроение, ракетостроение, автомобилестроение (характеристики кузова, работа двигателя, сопла);

— промышленная химия (разделение веществ, химические реакторы);

— метеорология, геология (потоки жидкости сквозь пористые среды, песчаники, дамбы);

— медицина (потоки крови, лимфы);

— кинематограф;

— моделирование физики для интерактивного обучения.

Процесс моделирования требует большого количества вычислительных ресурсов, поэтому долгое время симуляция в реальном времени была невозможна. Однако быстрое развитие компьютерной техники, в особенности графических процессоров и сопровождающего их ПО, позволяет приспособить и оптимизировать комплекс методов для выполнения этой задачи.

Целью выполнения курсового проекта является реализация физического симулятора поведения жидкости, её взаимодействия с объектами. Симулятор должен быть способным работать в режиме реального времени.

Для достижения поставленных целей ставились следующие задачи:

а) сбор материалов для реализации задачи;

б) анализ алгоритмов решения поставленной задачи;

в) разработка собственного алгоритма решения;

г) проведение экспериментов, стресс-тестов для разработанного решения.

1 Аналитический раздел

Гидро- и аэродинамика макроскопически описывается уравнениями Навье–Стокса. Они связывают давление, плотность и скорость жидкости в каждой точке пространства в каждый момент времени — в зависимости от начальных и граничных условий и параметров среды.

В случае вязкой слабосжимаемой жидкости система состоит из двух уравнений:

а) уравнения движения

$$\rho\left(\frac{\partial}{\partial t}+\mathbf{u}\cdot\nabla\right)\mathbf{u}=-\nabla p+\mu\Delta\mathbf{u}+\mathbf{f} \tag{1.1}$$

б) уравнения неразрывности

$$\nabla\cdot\mathbf{u}=0, \tag{1.2}$$

где ∇ — оператор набла;

Δ — векторный лапласиан;

ρ — плотность $\left[\frac{\text{кг}}{\text{м}^3}\right]$;

t — время, с;

$\mathbf{u}$ — векторное поле скоростей, $\left[\frac{\text{м}}{\text{с}}\right]$;

p — давление, Па;

μ — коэффициент динамической вязкости, $\left[\frac{\text{Н}\cdot\text{с}}{\text{м}^2}\right]$;

$\mathbf{f}$ — векторное поле внешних объёмных сил, $\left[\frac{\text{Н}}{\text{м}^3}\right]$.

Однако в аналитическом виде решения этих уравнений найдены лишь в некоторых частных случаях, в общем же случае показана невозможность решения данной «проблемы тысячелетия» существующими на настоящий момент средствами [1]. Поэтому нет полного понимания свойств уравнений Навье–Стокса, а значит численное моделирование остаётся основным способом исследования поведения жидкости.

Существует три фундаментальных подхода к данной задаче: сеточный метод, так называемый метод Эйлера, метод решёточных уравнений Больцмана и метод гидродинамики сглаженных частиц, известный как метод Лагранжа.

1.1 Метод решёточных уравнений Больцмана

Метод основан на дискретизации кинетического уравнения Больцмана, которое описывает, как меняется плотность распределения частиц по скоростям в каждой точке пространства со временем. Если проинтегрировать распределение частиц по скоростям в данной точке, то можно получить макроскопическую скорость для этой точки. Другими словами, макроскопически уравнение Больцмана эквивалентно уравнению Навье-Стокса.

Несмотря на то, что для плотных жидкостей уравнение Больцмана в общем виде не применимо, оно показывает неплохие результаты. В методе используется замена нижележащего уровня абстракций (микроскопическое уравнение для плотных жидкостей) физически неправильной реализацией (микроскопическое уравнение для разреженных газов), но так, что верхний уровень (макроскопическое уравнение Навье-Стокса) описывается верно.

Уравнение Больцмана, используемое в данном методе:

$$f\left(\mathbf{r}+\mathbf{v}dt, \mathbf{v}+\frac{\mathbf{F}}{m}dt, t+dt\right) - f(\mathbf{r},\mathbf{v},t) = -\frac{f-f_0}{\tau}dt \tag{1.3}$$

где f — функция распределения плотности вероятности частиц;

$\mathbf{r}$ — положение молекулы;

$\mathbf{v}$ — скорость молекулы;

t — время;

$\mathbf{F}$ — внешняя сила;

m — масса молекулы;

f_0 — равновесная функция распределения;

τ — время релаксации.

Чтоб получить возможность моделировать динамику непрерывного уравнения Больцмана на компьютере, необходимо его дискретизировать. Для этого вводится равномерная сетка пространственных координат с одинаковым шагом сетки по всем осям. Поведение жидкости определяется именно в узлах сетки. Фактически, разрешается молекулам находиться только в определенных пространственных узлах. Кроме того, нужно дискретизировать время, так как необходимо определять состояние жидкости в равноотстоящие

друг от друга моменты времени. Кроме того, молекулам позволяется иметь только определенные значения скорости — такие, что за шаг по времени они успевают перейти в какой-нибудь соседний узел. Эти разрешенные направления будут одинаковыми для всех пространственных узлов. Очевидно, что по диагональным направлениям скорости частиц будут больше, чем по недиагональным.

Интуитивно можно заключить, что при бесконечно малом шаге времени и шаге пространственной решетки эта дискретная система переходит в обычное уравнение Больцмана, которое, в свою очередь, переходит к уравнению Навье–Стокса в макроскопическом пределе.

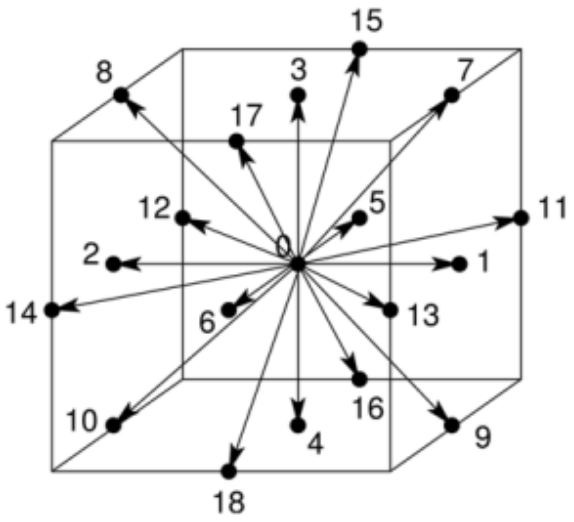

Рисунок 1.1 — Возможные направления движения частицы.

Таким образом, на каждом шаге вычислительной схемы необходимо «распространить» частицы, то есть переместить частицы, летевшие из одного узла в соседний (проделать это для всех частиц и направлений). После этого необходимо пересчитать массы, скорости, равновесные функции распределения. Наконец, необходимо «столкнуть» частицы, прилетевшие в данный узел, то есть перераспределить частицы по направлениям.

Преимущества метода:

— гибкость при задании граничных условий, поскольку такие условия задаются на макроскопическом уровне, а сам метод формулируется на микроскопическом уровне (потоки молекул);

— возможность моделировать поток смеси жидкостей или газов с разными параметрами;

— лёгкость распараллеливания.

Недостатки метода:

— сложность задания начальных условий: в начале моделирования необходимо задать потоки молекул из каждого узла по всем разрешенным направлениям;

— использование ограничено малыми скоростями;

— обладает неустойчивым поведением на границе подвижных тел.

1.2 Сеточный метод Эйлера

Для моделирования поведения жидкости необходимо иметь состояние в любой момент времени. Наиболее важной характеристикой для представления является скорость, потому что скорость определяет, как жидкость движется. Скорость изменяется во времени и пространстве, поэтому представляется в виде векторного поля, которое накладывается на дискретную сетку.

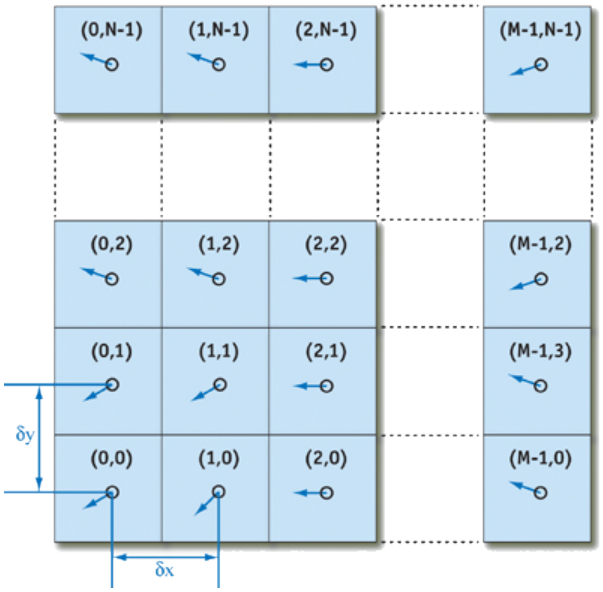

Рисунок 1.2 — Регулярная декартова сетка скоростей.

Для численного решения уравнений Навье–Стокса применяется несколько шагов для каждого компонента уравнений в отдельности и обновления значения скоростей в узлах сетки. Одним из самых важных шагов является применение разложения Гельмгольца, которое позволяет объединить

градиент давления ∇p и (1.2) в одно уравнение. Другие шаги используют явные, неявные и полуявные методы численного интегрирования. Использование сетки позволяет определять производные, используя метод конечных разностей, основанный на замене производных разностными схемами.

Преимущества метода:

— лучшее, по сравнению с другими методами, описание некоторых характеристик жидкости, например поля давлений и плотности.

Недостатки метода:

— динамичность жидкости ограничена сеткой;
— сложности при моделирования смеси жидкостей;
— большие затраты памяти для имитации высокого разрешения;
— сложность в распараллеливании.

1.3 Гидродинамика сглаженных частиц Лагранжа

Гидродинамика сглаженных частиц (ГСЧ) широко используется для моделирования поведения жидкости, газов, астрофизических феноменов и так далее.

ГСЧ является интерполяционным методом для приближения значений и производных непрерывного поля с помощью дискретных точек. Точки определены как сглаженные частицы, которые несут конкретные характеристики, например: массу, положение, плотность, температуру, давление, скорость и др. Характеристики частиц получаются как взвешенные из соседних частиц, то есть влияние каждой частицы на свойства оценивается в соответствии с её плотностью и расстоянием до интересующей частицы. Используемая при этом процедура интерполяции основывается на интегральной аппроксимации функций распределения в виде свёртки так называемым ядром сглаживания с последующей дискретизации вычисления данной свёртки по отдельным частицам.

На каждом шаге вычислительной схемы определяются силы, возникающие в результате вязкости и разности давлений, и внешние силы, после чего определяется ускорение каждой конкретной частицы и, наконец, скорости и координаты.

Преимущества метода:

— моделирование в условиях значительной деформации среды, свободными поверхностями и деформируемыми границами;

— возможность моделировать смеси жидкостей или газов с разными параметрами;

— гарантируется сохранение массы без дополнительных вычислений, так как частицы сами по себе представляют массу;

— лёгкость распараллеливания.

Недостатки метода:

— «смазывание» ударных волн и сильных контактных разрывов в большей степени, чем у современных сеточных методов;

— необходимо большое количество частиц для создания симуляции с эквивалентной разрешающей способностью сеточным методам.

2 Конструкторский раздел

2.1 Метод гидродинамики сглаженных частиц

2.1.1 Интегральная аппроксимация функции

Представление функций распределения физических величин в методе сглаженных частиц опирается на интегральное представление функции:

$$f(\mathbf{r}) = \int\limits_{\Omega} f(\tilde{\mathbf{r}})\delta(\mathbf{r} - \tilde{\mathbf{r}})d\tilde{\mathbf{r}}, \tag{2.1}$$

где f — непрерывная функция радиус-вектора $\mathbf{r}$;

Ω — объём, содержащий точку $\mathbf{r}$;

δ — дельта-функция Дирака:

$$\delta(\mathbf{r}) = \begin{cases} +\infty, & \mathbf{r} = 0, \\ 0, & \mathbf{r} \neq 0 \end{cases} \tag{2.2}$$

Первым приближением в методе сглаженных частиц является аппроксимация интегрального представления функции путём замены дельта-функции $\delta(\mathbf{r} - \tilde{\mathbf{r}})$ на функцию $\mathbb{W}(\mathbf{r} - \tilde{\mathbf{r}}, h)$, называемую ядром сглаживания:

$$f(\mathbf{r}) \approx \langle f(\mathbf{r}) \rangle = \int\limits_{\Omega} f(\tilde{\mathbf{r}})\mathbb{W}(\mathbf{r} - \tilde{\mathbf{r}}, h)d\tilde{\mathbf{r}}, \tag{2.3}$$

где $\langle\cdot\rangle$ — приближённое значение;

h — радиус сглаживания, определяющий область влияния ядра.

Ядро сглаживания должно аппроксимировать дельта-функцию Дирака, то есть

$$\lim_{h \to 0} \mathbb{W}(\mathbf{r} - \tilde{\mathbf{r}}, h) = \delta(\mathbf{r} - \tilde{\mathbf{r}}) \tag{2.4}$$

$$\mathbb{W}(\mathbf{r} - \tilde{\mathbf{r}}, h) \geqslant 0 \tag{2.5}$$

Рассмотрим погрешность данной аппроксимации. Для этого воспользуемся разложением в ряд Тейлора в окрестности точки $\mathbf{r}$ для дифференцируемой функции f:

$$f(\tilde{\mathbf{r}}) = f(\mathbf{r}) + \nabla f(\mathbf{r}) \cdot (\tilde{\mathbf{r}} - \mathbf{r}) + O(|\tilde{\mathbf{r}} - \mathbf{r}|^2)$$

Подставляя в (2.3), получим:

$$\begin{aligned}\langle f(\mathbf{r})\rangle &= \int\limits_{\Omega} \Big[f(\mathbf{r}) + \nabla f(\mathbf{r})\cdot(\tilde{\mathbf{r}}-\mathbf{r}) + O(|\tilde{\mathbf{r}}-\mathbf{r}|^2)\Big]\mathbb{W}(\mathbf{r}-\tilde{\mathbf{r}},h)d\tilde{\mathbf{r}} \\ &= f(\mathbf{r})\int\limits_{\Omega}\mathbb{W}(\mathbf{r}-\tilde{\mathbf{r}},h)d\tilde{\mathbf{r}} \\ &+ \nabla f(\mathbf{r})\cdot\int\limits_{\Omega}(\tilde{\mathbf{r}}-\mathbf{r})\mathbb{W}(\mathbf{r}-\tilde{\mathbf{r}},h)d\tilde{\mathbf{r}} \\ &+ \int\limits_{\Omega} O(|\tilde{\mathbf{r}}-\mathbf{r}|^2)\mathbb{W}(\mathbf{r}-\tilde{\mathbf{r}},h)d\tilde{\mathbf{r}}\end{aligned}$$

Для упрощения первого слагаемого введём условие нормировки

$$\int\limits_{\Omega}\mathbb{W}(\mathbf{r}-\tilde{\mathbf{r}},h)d\tilde{\mathbf{r}} = 1 \tag{2.6}$$

Для упрощения второго слагаемого введём условие чётности

$$\mathbb{W}(\mathbf{r}-\tilde{\mathbf{r}},h) = \mathbb{W}(\tilde{\mathbf{r}}-\mathbf{r},h) \tag{2.7}$$

Для упрощения третьего слагаемого введём условие финитности

$$\mathbb{W}(\mathbf{r}-\tilde{\mathbf{r}},h) = 0, \text{ для } |\mathbf{r}-\tilde{\mathbf{r}}| > h \tag{2.8}$$

И, следовательно,

$$\langle f(\mathbf{r})\rangle = f(\mathbf{r}) + O(h^2) \tag{2.9}$$

Таким образом, погрешность данной аппроксимации имеет второй порядок.

Сглаживающая функция $\mathbb{W}(\mathbf{r}-\tilde{\mathbf{r}},h)$ имеет большое значение, и от её выбора зависит аппроксимация решения в численном методе. Форма ядра задаёт степень взаимного влияния соседних частиц.

За время развития метода исследователями были предложены различные сглаживающие ядра, а также разработана методика получения подобных ядер [2].

2.1.2 Аппроксимация полей и их производных по частицам

Теперь, имея интегральную аппроксимацию функции, полученную с помощью сглаживающего ядра, можно перейти к аппроксимации на дискретном наборе частиц, на которые разбивается моделируемая среда, со своими дискретными значениями функции $f(\mathbf{r}_i)$.

В процессе подобной дискретизации интегралы могут быть заменены суммирование по соседним частицам, лежащим в носителе сглаживающей функции в данной точке. При этом бесконечно малый объём $d\tilde{\mathbf{r}}$ около положения j частицы заменяется на конечный объём частицы V_j и вводится её масса

$$m_j = V_j \rho_j, \tag{2.10}$$

где ρ_j — плотность частицы j.

Преобразуем уравнение приближённого значения:

$$\begin{aligned}
\langle f(\mathbf{r}) \rangle &= \int_{\Omega} f(\tilde{\mathbf{r}}) \mathbb{W}(\mathbf{r} - \tilde{\mathbf{r}}, h) d\tilde{\mathbf{r}} \\
&\approx \sum_{j=1}^{N} f(\mathbf{r}_j) \mathbb{W}(\mathbf{r} - \mathbf{r}_j, h) V_j \\
&= \sum_{j=1}^{N} f(\mathbf{r}_j) \mathbb{W}(\mathbf{r} - \mathbf{r}_j, h) \frac{m_j}{\rho_j},
\end{aligned}$$

где N — кол-во частиц в рассматриваемом носителе.

Таким образом, приближённое значение поля в j частице определяется как

$$f(\mathbf{r}_i) \approx \sum_{j=1}^{N} \frac{m_j}{\rho_j} f(\mathbf{r}_j) \mathbb{W}(\mathbf{r}_i - \mathbf{r}_j, h) \tag{2.11}$$

В уравнения, описывающие моделируемую среду (1.1), кроме самих физических величин входят также их пространственные первые и вторые производные, поэтому необходимо иметь возможность аппроксимировать градиенты и лапласианы этих функций.

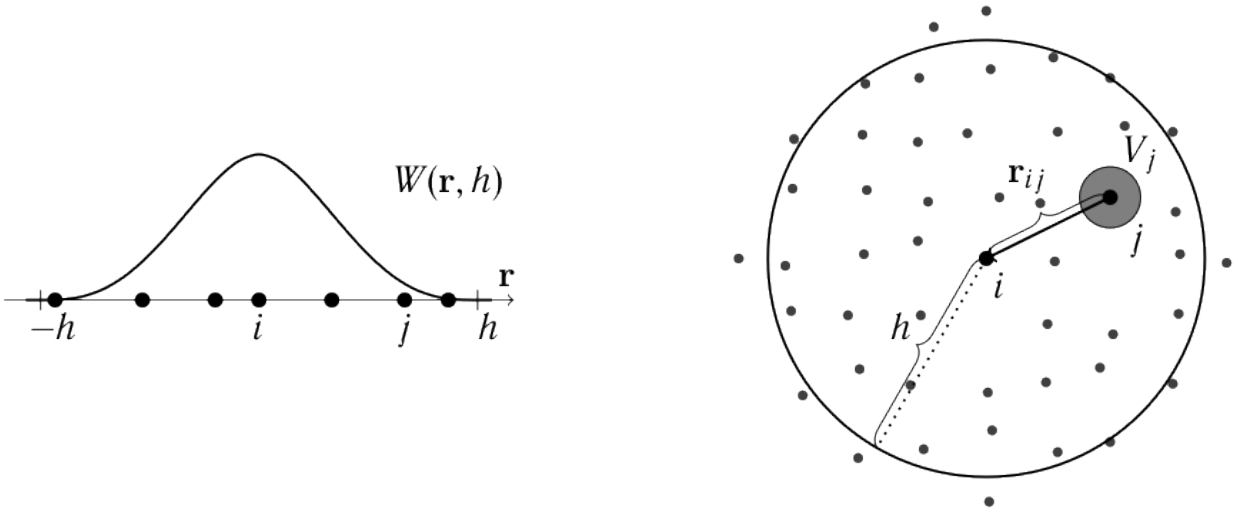

Рисунок 2.1 — Аппроксимация по частицам

$$\nabla\left(\frac{m_j}{\rho_j}f(\mathbf{r}_j)\mathbb{W}(\mathbf{r}_i-\mathbf{r}_j,h)\right)=\nabla\left(\frac{m_j}{\rho_j}f(\mathbf{r}_j)\right)\mathbb{W}(\mathbf{r}_i-\mathbf{r}_j,h)$$
$$+\frac{m_j}{\rho_j}f(\mathbf{r}_j)\nabla\left(\mathbb{W}(\mathbf{r}_i-\mathbf{r}_j,h)\right)$$
$$\cong 0\cdot\mathbb{W}(\mathbf{r}_i-\mathbf{r}_j,h)+\frac{m_j}{\rho_j}f(\mathbf{r}_j)\nabla\left(\mathbb{W}(\mathbf{r}_i-\mathbf{r}_j,h)\right),$$

где $\nabla\left(\frac{m_j}{\rho_j}f(\mathbf{r}_j)\right)\cong 0$, поскольку значение в точки j слабо зависит от координаты $\mathbf{r}$ и может быть принято постоянной величиной.

Таким образом, приближённое значение градиента поля в j частице

$$\nabla f(\mathbf{r}_i)\approx\sum_{j=1}^{N}\frac{m_j}{\rho_j}f(\mathbf{r}_j)\nabla\mathbb{W}(\mathbf{r}_i-\mathbf{r}_j,h) \tag{2.12}$$

Выражение для лапласиана записывается как

$$\Delta f(\mathbf{r}_i)\approx\sum_{j=1}^{N}\frac{m_j}{\rho_j}f(\mathbf{r}_j)\Delta\mathbb{W}(\mathbf{r}_i-\mathbf{r}_j,h) \tag{2.13}$$

Однако также используется «симметризованный» вариант аппроксимации градиента [3]:

$$\nabla f(\mathbf{r}_i)\approx\rho_i\left(\sum_{j=1}^{N}m_j\left(\frac{f(\mathbf{r}_j)}{\rho_j^2}+\frac{f(\mathbf{r}_i)}{\rho_i^2}\right)\nabla\mathbb{W}(\mathbf{r}_i-\mathbf{r}_j,h)\right) \tag{2.14}$$

Одним из важнейших отличий данных приближений является то, что значение функции f входят в них в форме парных частиц, что даёт более точные результаты при моделировании [3].

Получается, что представление в методе ГСЧ позволяет определить градиент поля по его значению и градиенту сглаживающей функции $\mathbb{W}$ вместо дифференцирования функции распределения поля.

2.1.3 Резюме

а) Метод ГСЧ является методом интерполяции, который может аппроксимировать непрерывные поля и их производные с помощью дискретных точек, называемых частицами.

б) Частицы обладают массой m, позицией $\mathbf{r}$ и скоростью $\mathbf{u}$, но могут также содержать расчётные величины, например плотность ρ, давление p и так далее.

в) Связь массы, плотности и объёма определяется как $V = \frac{m}{\rho}$.

г) Сглаживающая функция должна:

1) аппроксимировать дельта-функцию: $lim_{h\to 0}\mathbb{W}(\mathbf{r}-\tilde{\mathbf{r}}, h) = \delta(\mathbf{r}-\tilde{\mathbf{r}})$
2) быть неотрицательной: $\mathbb{W}(\mathbf{r}-\tilde{\mathbf{r}}, h) \geqslant 0$
3) быть нормированной: $\int_\Omega \mathbb{W}(\mathbf{r}-\tilde{\mathbf{r}}, h)d\tilde{\mathbf{r}} = 1$
4) быть чётной: $\mathbb{W}(\mathbf{r}-\tilde{\mathbf{r}}, h) = \mathbb{W}(\tilde{\mathbf{r}}-\mathbf{r}, h)$
5) обладать финитностью: $\mathbb{W}(\mathbf{r}-\tilde{\mathbf{r}}, h) = 0,$ для $|\mathbf{r}-\tilde{\mathbf{r}}| > h$

д) Формулы аппроксимации для частиц:

1) $f(\mathbf{r}_i) \approx \sum_{j=1}^{N} \frac{m_j}{\rho_j} f(\mathbf{r}_j)\mathbb{W}(\mathbf{r}_i - \mathbf{r}_j, h)$
2) $\nabla f(\mathbf{r}_i) \approx \sum_{j=1}^{N} \frac{m_j}{\rho_j} f(\mathbf{r}_j)\nabla\mathbb{W}(\mathbf{r}_i - \mathbf{r}_j, h)$
3) $\Delta f(\mathbf{r}_i) \approx \sum_{j=1}^{N} \frac{m_j}{\rho_j} f(\mathbf{r}_j)\Delta\mathbb{W}(\mathbf{r}_i - \mathbf{r}_j, h)$
4) $\nabla f(\mathbf{r}_i) \approx \rho_i\Big(\sum_{j=1}^{N} m_j\big(\frac{f(\mathbf{r}_j)}{\rho_j^2} + \frac{f(\mathbf{r}_i)}{\rho_i^2}\big)\nabla\mathbb{W}(\mathbf{r}_i - \mathbf{r}_j, h)\Big)$

е) ГСЧ применим для задач с сжимаемыми жидкостями и газами.

2.2 Гидродинамика Лагранжа

Использование метода частиц вместо сеток позволяет избавится от уравнения неразрывности (1.2), так как количество частиц фиксировано на

время моделирования, а масса закреплена за каждой частицей, то уравнение выполняется автоматически.

Заметим, что

$$\begin{aligned}\frac{d}{dt}\mathbf{u}\big(t,\mathbf{r}(t)\big) &= \frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{u}}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial \mathbf{u}}{\partial y}\frac{\partial y}{\partial t} + \frac{\partial \mathbf{u}}{\partial z}\frac{\partial z}{\partial t} \\ &= \frac{\partial \mathbf{u}}{\partial t} + \left(\frac{\partial \mathbf{u}}{\partial x}, \frac{\partial \mathbf{u}}{\partial y}, \frac{\partial \mathbf{u}}{\partial z}\right) \cdot \left(\frac{\partial x}{\partial t}, \frac{\partial y}{\partial t}, \frac{\partial z}{\partial t}\right) \\ &= \frac{\partial \mathbf{u}}{\partial t} + \left(\mathbf{u} \cdot \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}\right)\right)\mathbf{u} \\ &= \left(\frac{\partial}{\partial t} + \mathbf{u} \cdot \nabla\right)\mathbf{u}\end{aligned} \tag{2.15}$$

А, значит, уравнение движения (1.1) можно записать как

$$\rho\frac{d\mathbf{u}}{dt} = -\nabla p + \mu\Delta\mathbf{u} + \mathbf{f} \tag{2.16}$$

Правая часть уравнения состоит из внутренних и внешних объёмных сил, $\mathbf{F} = -\nabla p + \mu\Delta\mathbf{u} + \mathbf{f}$. Для частицы i ускорение принимает вид

$$\mathbf{a}_i = \frac{d\mathbf{u}_i}{dt} = \frac{\mathbf{F}_i}{\rho_i} \tag{2.17}$$

Если не указано другое, то в качестве сглаживающего ядра будем использовать предложенное в [4]

$$\mathbb{W}_d(\mathbf{r}, h) = \frac{315}{64\pi h^9}\begin{cases}\left(h^2 - |\mathbf{r}|^2\right)^3, & 0 \leqslant |\mathbf{r}| \leqslant h, \\ 0, & h < |\mathbf{r}|\end{cases} \tag{2.18}$$

2.2.1 Плотность

Согласно ГСЧ любое непрерывное значение поля, его градиент и лапласиан могут быть аппроксимированы, используя уравнения (2.11), (2.12) и (2.13) соответственно. В уравнениях предполагается, что массы и плотности известны для всех частиц. Так как масса задаётся пользователем, то необходимо предварительно найти только плотности.

Для этого воспользуемся уравнением (2.11) и получим окончательное выражение:

$$\rho_i = \sum_j m_j \mathbb{W}_d(\mathbf{r}_i - \mathbf{r}_j, h) \tag{2.19}$$

2.2.2 Давление

Давление частицы может быть определено, используя уравнение идеального газа

$$pV = \nu RT, \tag{2.20}$$

где V — объём, $V = \frac{m}{\rho}$;

ν — количество вещества;

R — универсальная газовая постоянная;

T — температура.

Так как мы рассматривает изотермическую жидкость с постоянной массой, то введём константу $k = \frac{\nu}{m} RT$:

$$\frac{p}{\rho} = k \Rightarrow p = k\rho$$

Однако такой подход приводит к чисто отталкивающим силам между частицами, что справедливо для идеального газа, который имеет тенденцию к расширению в пространстве. В противоположность этому, жидкости обладают внутренней связностью и имеют постоянную плотность в покое. В [5] предлагается использовать модифицированную версию уравнения состояния идеального газа:

$$(p + p_0)V = k$$
$$p + k\rho_0 = k\rho$$

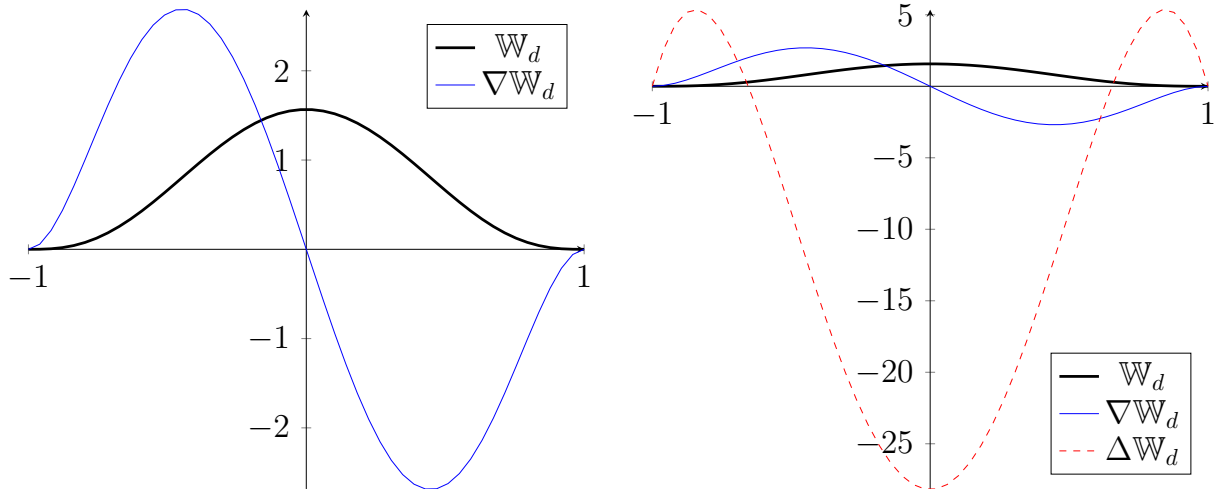

Рисунок 2.2 — Стандартное сглаживающее ядро при $h = 1$.

$$p = k(\rho - \rho_0), \tag{2.21}$$

где ρ_0 — плотность жидкости в равновесном состоянии.

Так как речь идёт о силе, то использовать стандартное уравнение для аппроксимации градиента функции (2.12) недостаточно, так как оно асимметрично, а значит третий закон Ньютона не выполняется. Поэтому мы будем использовать симметричное уравнение (2.14):

$$\mathbf{f}_i^p = -\nabla p(\mathbf{r}_i) = -\rho_i \sum_{j \neq i} \left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2} \right) m_j \nabla \mathbb{W}(\mathbf{r}_i - \mathbf{r}_j, h).$$

Использование $\mathbb{W} = \mathbb{W}_d$ в данном случае является не лучшим выбором, поскольку $\nabla \mathbb{W}_d(\mathbf{r}, h) \to 0$ при $|\mathbf{r}| \to 0$ (рис. 2.2), что ведёт к кластеризации частиц в регионах с высоким давлением. Поэтому в [5] была предложена другая сглаживающая функция:

$$\mathbb{W}_p(\mathbf{r}, h) = \frac{15}{\pi h^6} \begin{cases} (h - |\mathbf{r}|)^3, & 0 \leqslant |\mathbf{r}| \leqslant h \\ 0, & h < |\mathbf{r}| \end{cases} \tag{2.22}$$

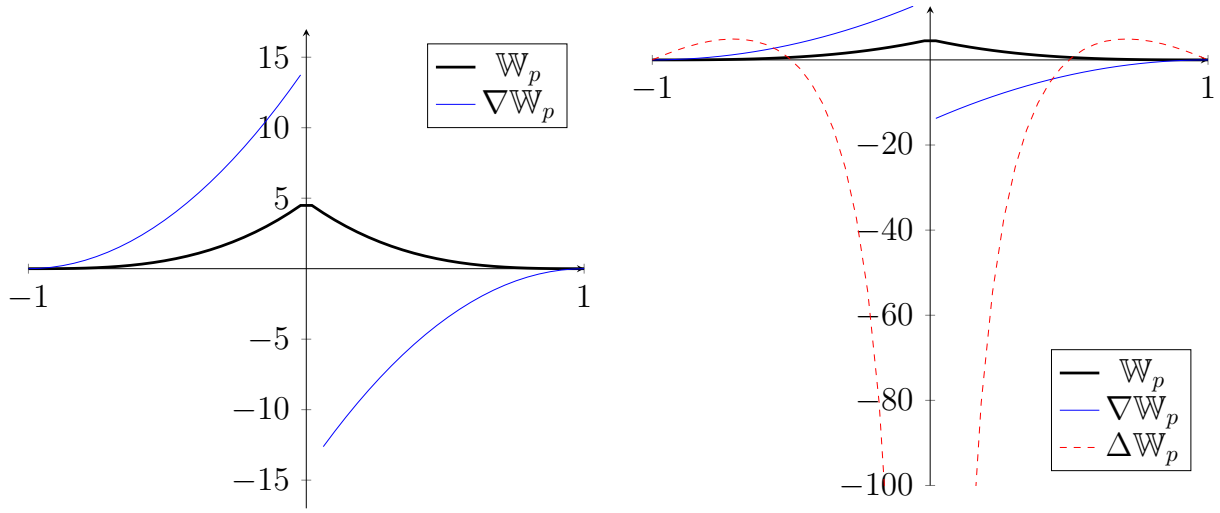

Рисунок 2.3 — Сглаживающее ядро для давления при $h = 1$.

Градиент которой выражается как

$$\begin{gathered} \nabla \mathbb{W}_p(\mathbf{r}, h) = -\frac{45}{\pi h^6} \frac{\mathbf{r}}{|\mathbf{r}|} (h - |\mathbf{r}|)^2 \\ \lim_{r \to 0 \mp 0} \nabla \mathbb{W}_p(r, h) = \pm \frac{45}{\pi h^4} \end{gathered} \tag{2.23}$$

Что позволяет моделировать отталкивание в областях высокой плотности и притягивание в областях низкой плотности (рис. 2.4).

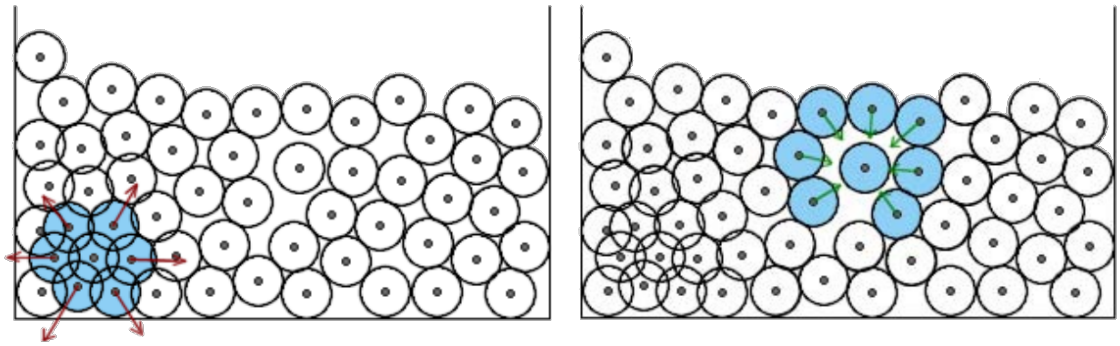

Рисунок 2.4 — Отталкивание и притягивание в зависимости от плотности.

Окончательное уравнение:

$$\mathbf{f}_i^p = -\rho_i \sum_{j \neq i} \left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2} \right) m_j \nabla \mathbb{W}_p(\mathbf{r}_i - \mathbf{r}_j, h) \tag{2.24}$$

2.2.3 Вязкость

Жидкость представляет собой вещество, которое не может противостоять напряжению сдвига и, следовательно, будет течь при деформации. В то же время, когда жидкость течет, молекулы испытывают внутреннее трение, а значит часть кинетической энергии переходит в тепло. Сопротивление текучести называется вязкостью, а соответствующая сила входит в (2.16) в виде $\mathbf{f}_i^v = \mu \Delta \mathbf{u}(\mathbf{r}_i)$, где μ — динамическая вязкость, которая уменьшается с увеличением температуры, и растёт с увеличением давления.

Поскольку силы вязкости зависят только от разности скоростей, а не от их абсолютных значений, то простейший способ симметризовать аппроксимацию — использовать разность скоростей:

$$\mathbf{f}_i^v = \mu \Delta \mathbf{u}(\mathbf{r}_i) = \mu \sum_j (\mathbf{u}_j - \mathbf{u}_i) \frac{m_j}{\rho_j} \Delta \mathbb{W}(\mathbf{r}_i - \mathbf{r}_j, h)$$

Возможная интерпретация формулы: частица i ускоряется в направлении относительного движения среды.

Заметим, что лапласиан сглаживающего ядра должен был всюду положительным, то есть $\Delta \mathbb{W}(\mathbf{r}, h) \geqslant 0$ при $|\mathbf{r}| \leqslant h$. Это необходимо, потому что мы не хотим, чтобы силы, вызванные вязкостью увеличивали относительную скорость, и таким образом вводили нестабильность в систему.

Получается, что стандартное сглаживающее ядро ($\mathbb{W}_d$) и ядро для давления ($\mathbb{W}_p$) не подходят (рис. 2.3). В [4] предложено использовать следующее ядро:

$$\mathbb{W}_v(\mathbf{r}, h) = \frac{15}{2\pi h^3} \begin{cases} -\frac{|\mathbf{r}|^3}{2h^3} + \frac{|\mathbf{r}|^2}{h^2} + \frac{h}{2|\mathbf{r}|} - 1, & 0 < |\mathbf{r}| \leqslant h \\ 0, & h < |\mathbf{r}| \end{cases} \tag{2.25}$$

$$\lim_{r \to 0} \mathbb{W}_v(r, h) = \infty$$

Лапласиан которого выражается как

$$\Delta \mathbb{W}_v(\mathbf{r}, h) = \frac{45}{\pi h^6}(h - |\mathbf{r}|) \tag{2.26}$$

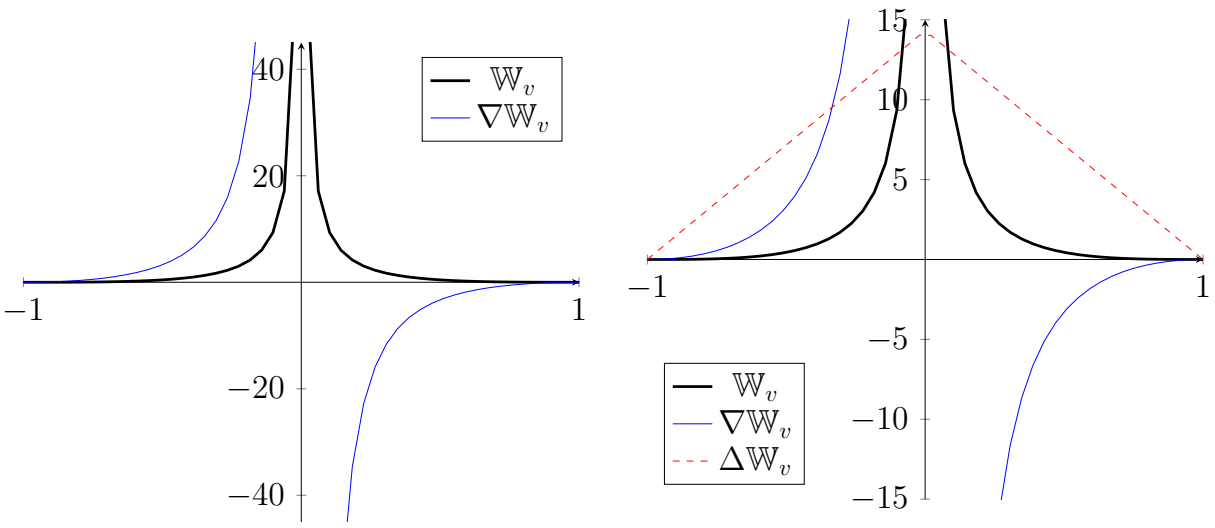

Рисунок 2.5 — Сглаживающее ядро для вязкости при $h = 1$.

Конечное выражение для сил вязкости:

$$\mathbf{f}_i^v = \mu \sum_j (\mathbf{u}_j - \mathbf{u}_i) \frac{m_j}{\rho_j} \Delta \mathbb{W}_v(\mathbf{r}_i - \mathbf{r}_j, h) \tag{2.27}$$

2.2.4 Внешние силы

Существуют различные внешние силы, подлежащие моделированию (например, поверхностное натяжение и гравитация). Более того, так как метод является интерактивным, внешние силовые поля также могут быть введены и удалены динамически во время моделирования.

В уравнении (2.16) внешние силы — последнее слагаемое, $\mathbf{f}$, которое может быть определено как сумма всех полей внешних объёмных сил:

$$\mathbf{f} = \sum_n \mathbf{f}^n \tag{2.28}$$

Обработка столкновений, которые также могут рассматривается посредством внешний сил, рассматриваются в следующем разделе отдельно.

2.2.4.1 Гравитация

Гравитация действуют на все частицы:

$$\mathbf{f}_i^g = \rho_i \mathbf{g}, \tag{2.29}$$

где $\mathbf{g}$ — ускорение свободного падения.

2.2.4.2 Поверхностное натяжение

Силы поверхностного натяжения являются внешними силами, применяемыми к свободной поверхности жидкой среды. Они, как правило, не является частью уравнений Навье-Стокса, так как считаются граничным условием. Молекулы жидкости находятся под влиянием сил притяжения от соседних молекул, которые уравновешены внутри жидкости. Но на поверхности силы не сбалансированы и вызывают поверхностное натяжение. Эти силы сонаправлены со внутренней нормалью к поверхности жидкости. Силы поверхностного натяжения стремятся сгладить кривизну поверхности (рис. 2.6).

Модель для расчёта сил поверхностного натяжения строится на «цветовой» функции [4]:

$$c(\mathbf{r}) = \begin{cases} 1, & \exists i : \mathbf{r} = \mathbf{r}_i \\ 0 & \text{иначе} \end{cases}$$

Таким образом,

$$c_i = c(\mathbf{r}_i) = \sum_j \frac{m_j}{\rho_j} \mathbb{W}_d(\mathbf{r}_i - \mathbf{r}_j, h) \tag{2.30}$$

При этом, внутренняя нормаль вычисляется как

$$\mathbf{n}_i = \nabla c(\mathbf{r}_i) = \sum_j \frac{m_j}{\rho_j} \nabla \mathbb{W}_d(\mathbf{r}_i - \mathbf{r}_j, h) \tag{2.31}$$

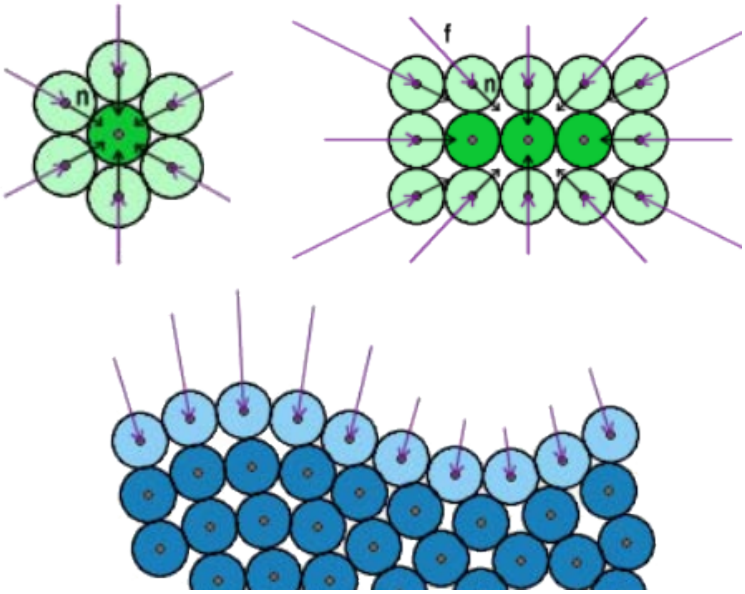

Рисунок 2.6 — Силы поверхностного натяжения.

Тогда сила поверхностного натяжения принимает вид

$$\mathbf{f}_i^s = -\sigma \Delta c_i \frac{\mathbf{n}_i}{|\mathbf{n}_i|}, \tag{2.32}$$

где σ — коэффициент поверхностного натяжения.

Так как $\frac{\mathbf{n}_i}{|\mathbf{n}_i|}$ численно нестабильно при $|\mathbf{n}_i| \to 0$, необходимо, чтобы частица находилась рядом с поверхностью:

$$|\mathbf{n}_i| \geqslant l, \tag{2.33}$$

где $l > 0$ — некоторая константа предела применения.

2.2.5 Обработка столкновений

Когда частицы сталкиваются с контейнером, они должны оставаться внутри своих границ. Аналогичным образом, если частицы сталкиваются с препятствием, они не могут проникнуть или получить доступ к внутренней части объекта.

Обработка столкновений может быть разделена на два этапа: обнаружение столкновения и реакция на столкновение. В данной работе все препятствия и контейнеры считаются фиксированными и жесткими, следовательно мы будем рассматривать только реакцию на частицы.

2.2.5.1 Определение столкновений

Частицы несут информацию о положении $\mathbf{x}$ и скорости, которых достаточно для определения коллизий. Если текущее положение частицы не даёт достаточно информации, чтобы обеспечить правильное соударение, то скорость может быть использована для получения прежней позиции частицы. Обязательная информация о столкновении включает в себя:

а) Точка контакта с поверхностью, $\mathbf{cp}$.
б) Глубина проникновения, d.
в) Нормаль к поверхности в точке контакта, $\mathbf{n}$.

Заметим, что глубина проникновения положительна ($d > 0$), то есть нахождение частицы на поверхности объекта не ведёт к коллизии.

Рассмотрим два возможных определения столкновений (рис. 2.7): корректное ($\mathbf{cp}_1$) и упрощённое ($\mathbf{cp}_2$). Первый более точный, однако последний менее требовательный к вычислительным ресурсам. На практике использование упрощённого варианта не оказывает видимого влияния на симуляцию [6], так как при небольших временных интервалов (рассматриваются в следующей разделе) $d_1 \to 0$, а значит и $\mathbf{cp}_2 \to \mathbf{cp}_1$. Исключение из этого правила составляют небольшие по объёму объекты, однако они в данной работе не рассматриваются.

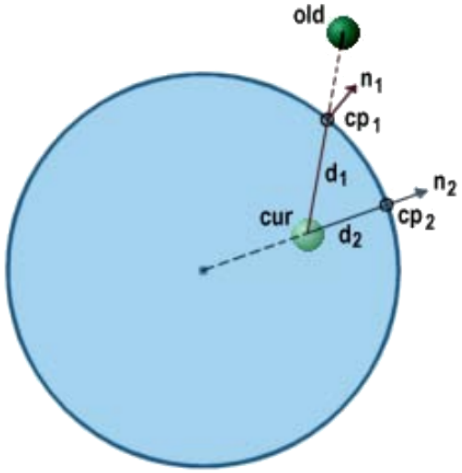

Рисунок 2.7 — Два возможных способа определения столкновений.

Введём функцию, позволяющую определять положение частицы относительно рассматриваемого объекта:

$$F(\mathbf{x}) \begin{cases} < 0, & \text{частица внутри объекта} \\ = 0, & \text{частица на поверхности} \\ > 0, & \text{частица вне объекта} \end{cases} \tag{2.34}$$

2.2.5.2 Сфера

Тест на пересечение для сферы записывается как

$$F_s(\mathbf{x}) = |\mathbf{x} - \mathbf{c}|^2 - r^2, \tag{2.35}$$

где $\mathbf{c}$ — центр сферы;
r — радиус сферы.

Если коллизия имеет место, то точка контакта, глубина проникновения и нормаль к поверхности вычисляется соответственно

$$\mathbf{cp}_s = \mathbf{c} + r\frac{\mathbf{x} - \mathbf{c}}{|\mathbf{x} - \mathbf{c}|} \tag{2.36}$$

$$d_s = \big||\mathbf{c} - \mathbf{x}| - r\big| \tag{2.37}$$

$$\mathbf{n}_s = \operatorname{sgn}\big(F_s(\mathbf{x})\big)\frac{\mathbf{c} - \mathbf{x}}{|\mathbf{c} - \mathbf{x}|} \tag{2.38}$$

2.2.5.3 Ограничивающий прямоугольный параллелепипед

Введём два вспомогательных векторных оператора:

$$[\mathbf{a}]_{abs} := (|a_x|, |a_y|, |a_z|),$$
$$[\mathbf{a}]_{max} := \max(a_x, a_y, a_z)$$

Тогда тест на пересечение для ограничивающего прямоугольного параллелепипеда определяется как

$$F_b(\mathbf{x}) = \big[[\mathbf{x}']_{abs} - \mathbf{e}\big]_{max}, \tag{2.39}$$

где $\mathbf{x}'$ — координаты точки $\mathbf{x}$ в системе координат параллелепипеда.

$$\mathbf{x}' = (\mathbf{x} - \mathbf{c})\mathbf{S}^{-1}, \tag{2.40}$$

где $\mathbf{S}$ — матрица перехода от мирового базиса к локальному; $\mathbf{c}$ — центр параллелепипеда в мировой системе координат.

Точка пересечения в с.к. параллелепипеда может быть найдена как

$$\mathbf{cp}'_b = \min\big[\mathbf{e}, \max[-\mathbf{e}, \mathbf{x}']\big], \tag{2.41}$$

а в мировой с.к.

$$\mathbf{cp}_b = \mathbf{c} + \mathbf{cp}'_b\mathbf{S} \tag{2.42}$$

Тогда глубина проникновения и нормаль:

$$d_b = |\mathbf{cp}_b - \mathbf{x}| \tag{2.43}$$

$$\mathbf{n}_b = \frac{\operatorname{sgn}[\mathbf{cp}'_b - \mathbf{x}']\mathbf{S}}{\big|\operatorname{sgn}[\mathbf{cp}'_b - \mathbf{x}']\mathbf{S}\big|} \tag{2.44}$$

2.2.5.4 Реакция на столкновение

Когда столкновение обнаружено, необходимо скорректировать позицию частицы и её скорость. В нашей модели, если частица i проникла через препятствие, то её новая позиция может быть определена как

$$\mathbf{r}_i = \mathbf{r}_i + d\mathbf{n} = \mathbf{cp} \tag{2.45}$$

Вектор скорости может быть скорректирован согласно отражению для частично упругого столкновения:

$$\mathbf{u}_i = \mathbf{u}_i - (1 + \beta)(\mathbf{u}_i \cdot \mathbf{n})\mathbf{n},$$

где β — коэффициент восстановления, $0 \leqslant \beta \leqslant 1$.

Однако данная модель не идеальна. Так как при столкновении мы имеем дело с частицей, которая уже находится внутри препятствия, то данное уравнение может неправомерно увеличить кинетическую энергию частицы. Мы стремимся моделировать столкновение частицы с импульсом точно по времени, когда произошло столкновение. В [6] предлагается использовать для этого временную корректировку:

$$\mathbf{u}_i = \mathbf{u}_i - (1 + \beta\frac{d}{\Delta t|\mathbf{u}_i|})(\mathbf{u}_i \cdot \mathbf{n})\mathbf{n} \tag{2.46}$$

2.2.6 Численное интегрирование времени

Во время моделирования используется глобальное фиксированное интервал Δt. После вычисления ускорения (2.17), позиция частиц может быть получена интегрированием ускорения. Согласно [6], для гидродинамики сглаженных частиц лучше всего показывает себя метод «чехарды со средней точкой» (рис. 2.8):

$$\mathbf{u}_{t+\frac{1}{2}\Delta t} = \mathbf{u}_{t-\frac{1}{2}\Delta t} + \Delta t \mathbf{a}_t, \tag{2.47}$$

$$\mathbf{r}_{t+\Delta t} = \mathbf{r}_t + \Delta t \mathbf{u}_{t+\frac{1}{2}\Delta t}, \tag{2.48}$$

с начальным смещением скорости

$$\mathbf{u}_{-\frac{1}{2}\Delta t} = \mathbf{u}_0 - \frac{1}{2}\Delta t \mathbf{a}_0 \tag{2.49}$$

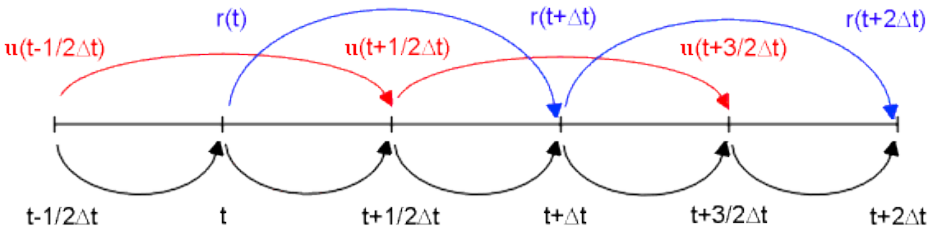

Рисунок 2.8 — Метод «чехарды со средней точкой».

Скорость для времени t может быть получена как

$$\mathbf{u}_t \approx \frac{\mathbf{u}_{t-\frac{1}{2}\Delta t} + \mathbf{u}_{t+\frac{1}{2}\Delta t}}{2}, \tag{2.50}$$

которое требуется при расчёте сил для времени t.

2.2.7 Резюме

а) Ускорение частицы под действием объёмных сил $\mathbf{a}_i = \frac{d\mathbf{u}_i}{dt} = \frac{\mathbf{F}_i}{\rho_i}$

б) Плотность частицы $\rho_i = \sum_j m_j \mathbb{W}_d(\mathbf{r}_i - \mathbf{r}_j, h)$

в) Внутренняя нормаль к поверхности: $\mathbf{n}_i = \sum_j \frac{m_j}{\rho_j} \nabla \mathbb{W}_d(\mathbf{r}_i - \mathbf{r}_j, h)$

г) Объёмные силы, действующие на частицу, возникают под действием

1) давления $\mathbf{f}_i^p = -\rho_i \sum_{j \neq i} \left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2} \right) m_j \nabla \mathbb{W}_p(\mathbf{r}_i - \mathbf{r}_j, h)$
2) вязкости $\mathbf{f}_i^v = \mu \sum_j (\mathbf{u}_j - \mathbf{u}_i) \frac{m_j}{\rho_j} \Delta \mathbb{W}_v(\mathbf{r}_i - \mathbf{r}_j, h)$
3) гравитации $\mathbf{f}_i^g = \rho_i \mathbf{g}$

4) поверхностного натяжения $\mathbf{f}_i^s = -\sigma \frac{\mathbf{n}_i}{|\mathbf{n}_i|} \sum_j \frac{m_j}{\rho_j} \Delta \mathbb{W}_d(\mathbf{r}_i - \mathbf{r}_j, h)$

д) Для обработки столкновения частицы со сферой необходимы

1) тестовая функция $F_s(\mathbf{x}) = |\mathbf{x} - \mathbf{c}|^2 - r^2$
2) точка контакта $\mathbf{cp}_s = \mathbf{c} + r \frac{\mathbf{x}-\mathbf{c}}{|\mathbf{x}-\mathbf{c}|}$
3) глубина проникновения $d_s = \big||\mathbf{c} - \mathbf{x}| - r\big|$
4) нормаль $\mathbf{n}_s = \operatorname{sgn}\big(F_s(\mathbf{x})\big) \frac{\mathbf{c}-\mathbf{x}}{|\mathbf{c}-\mathbf{x}|}$

е) Для обработки столкновения с параллелепипедом необходимы

1) тестовая функция $F_b(\mathbf{x}) = \big[[\mathbf{x}']_{abs} - \mathbf{e}\big]_{max}$
2) точка контакта $\mathbf{cp}_b = \mathbf{c} + \mathbf{cp}'_b\mathbf{S}, \quad \mathbf{cp}'_b = \min\big[\mathbf{e}, \max[-\mathbf{e}, \mathbf{x}']\big]$
3) глубина проникновения $d_b = |\mathbf{cp}_b - \mathbf{x}|$
4) нормаль $\mathbf{n}_b = \frac{\operatorname{sgn}[\mathbf{cp}'_b - \mathbf{x}']\mathbf{S}}{\big|\operatorname{sgn}[\mathbf{cp}'_b - \mathbf{x}']\mathbf{S}\big|}, \quad \mathbf{x}' = (\mathbf{x} - \mathbf{c})\mathbf{S}^{-1}$

ж) Реакция на столкновение осуществляется путём корректировки

1) положения $\mathbf{r}_i = \mathbf{cp}$
2) скорости $\mathbf{u}_i = \mathbf{u}_i - (1 + \beta \frac{d}{\Delta t |\mathbf{u}_i|})(\mathbf{u}_i \cdot \mathbf{n})\mathbf{n}$

з) Для вычисления положения и скорости используется метод «чехарды со средней точкой»:

1) новая скорость $\mathbf{u}_{t+\frac{1}{2}\Delta t} = \mathbf{u}_{t-\frac{1}{2}\Delta t} + \Delta t \mathbf{a}_t$
2) новое положение $\mathbf{r}_{t+\Delta t} = \mathbf{r}_t + \Delta t \mathbf{u}_{t+\frac{1}{2}\Delta t}$
3) начальное смещение скорости $\mathbf{u}_{-\frac{1}{2}\Delta t} = \mathbf{u}_0 - \frac{1}{2}\Delta t \mathbf{a}_0$
4) текущая скорость $\mathbf{u}_t \approx \frac{\mathbf{u}_{t-\frac{1}{2}\Delta t} + \mathbf{u}_{t+\frac{1}{2}\Delta t}}{2}$

2.3 Триангуляция

2.3.1 Шагающие кубики

Часто в компьютерной графике приходится сталкиваться с визуализацией элементов существующих абстрактно, с визуальной точки зрения, но реально не видимых. Форма и атрибуты таких элементов задаются посредством не геометрических абстракций, а основываясь на других предметных областях, например, на физических или биологических свойствах отображаемого объекта.

Алгоритм «шагающих кубиков» (marching cubes) разработан с целью визуализации изоповерхностей посредством трехмерных скалярных полей. В

методе обеспечивается высокая скорость и качество визуализации, что позволяет использовать подобные методы в приложениях реального времени.

Предполагается, что пространство представимо в виде трехмерной сетки, то есть представляет из себя набор вокселей, которые являются минимальными структурными единицами этого пространства.

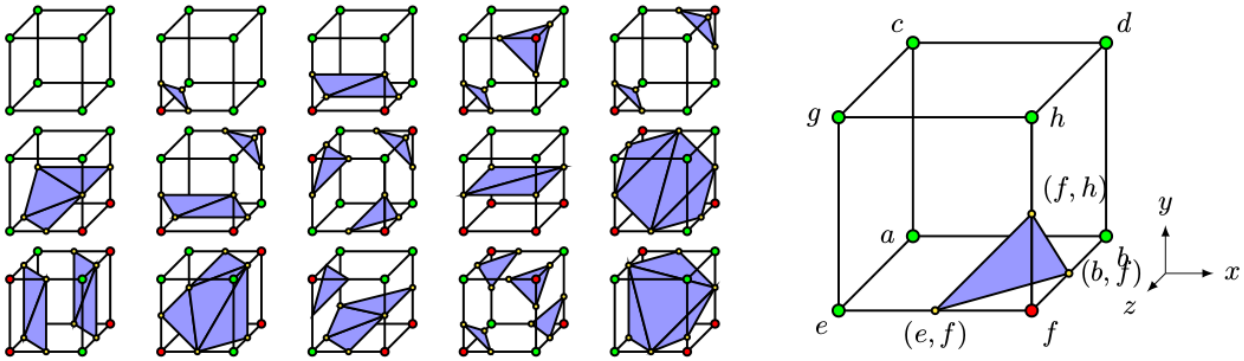

Рисунок 2.9 — Базовые предопределённые треугольники.

Данный метод рассматривает все возможные комбинации точек со значением больше и со значением меньше заданного. С учетом симметрии в [7] предложено, что различных возможных базовых комбинаций всего 15 (рис. 2.9), а всего комбинаций — 256. Таким образом, мы заранее знаем набор примитивов (в данном случае, треугольников) которые будут отображаться. Следовательно мы можем заранее просчитать эти примитивы, а также их атрибуты (нормали, цвета, и так далее).

Размер сетки является первым признаком улучшения качества и точности получаемой изоповерхности. Чем меньше выбирается шаг сетки, тем больше получается вокселей, тем более гладкой и точной получается результирующая изоповерхность, но тем медленнее она будет обрабатываться и рисоваться.

Понятно, что большой набор возможных комбинаций (256) ставит задачу продуманной работы с такими большими массивами данных без потери производительности и с достаточной степенью гибкости для возможности дальнейших оптимизаций. Рассмотрим основные оптимизации.

Для начала, основываясь на принятой нами индексации вершин, составляется идентификатор комбинации, который может принимать значение от 0 до 255 (по числу возможных вариантов). Так как каждая вершина может

находится только в двух состояниях, то считается он следующим образом:

$$I = \prod_{i=0}^{7} 2^i \cdot s(p_i, l), \quad s(x, y) = \begin{cases} 1, & x \leqslant y \\ 0 & \text{иначе} \end{cases}, \tag{2.51}$$

где p_i — значение в вершине i;

l — заданный уровень изоповерхности.

Далее строится таблица, которая для каждого индекса, подсчитанного указанным выше способом, определяет те ребра, которые будет пересекать изоповерхность. Так как ребер всего 12, то данная таблица содержит 12-битные числа, подсчитанные аналогичным способом. Данную таблицу необходимо использовать для того, чтобы определить позицию вершины очередного треугольника.

Теперь, зная ребро, на которой будет находится вершина треугольника, вычислим её координаты. Как уже говорилось выше, позиция точки зависит от значений потенциала в вершинах и значения уровня изоповерхности:

$$\mathbf{v} = \mathbf{v}_i + (l - p_i)\frac{\mathbf{v}_j - \mathbf{v}_i}{p_j - p_i}, \tag{2.52}$$

где i и j — вершины, связанные ребром;

$\mathbf{v}_i$ и $\mathbf{v}_j$ — координаты вершин i и j.

Заметим, что если не проводить интерполяцию, а использовать за искомое значение, например, середину ребра, то все возможные треугольники могут быть просчитаны заранее, и, как следствие, могут быть просчитаны заранее нормали и другие атрибуты для них, следовательно мы избавляемся от большой работы, которую нам иначе бы пришлось выполнять в реальном времени. Однако, приходится мириться с потерей качества. Следовательно, в приложениях, где требуется высокая точность построения, как в данной работе, необходимо проводить интерполяцию, а в приложениях, где построение изоповерхности используется лишь как дополнительный элемент визуализации, достаточно грубого, но быстрого построения изоповерхности с полным предвычислением геометрии.

2.3.2 Гистограммные пирамиды

Каждый воксель может порождать до четырёх треугольников, что ведёт к большому расходу памяти, часто превышающего ограничения GPU. Дело усложняется тем, что во время рендеринга необходимо обойти все воксели, так как любой из них мог дать треугольники, что очень неэффективно.

Для решения данной проблемы используется подход гистопирамид, позволяя получить список активных вокселей (то есть порождающих треугольники) из сильно разреженной сетки вокселей [8].

На первом шаге метода производится последовательная свертка матрицы рассматриваемых вокселей в четыре раза на каждой итерации, создавая квадродеревья для последующего обхода (рис. 2.10). Также в результате свертки мы имеем количество активных вокселей.

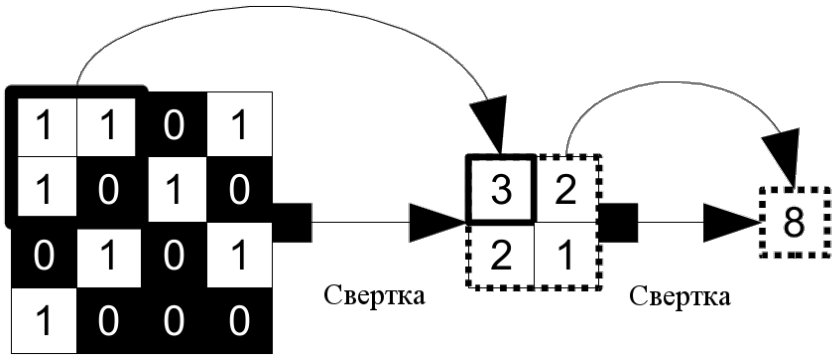

Рисунок 2.10 — Создание гистограммной пирамиды

Следующим шагом создаётся непосредственно список активных вокселей. Для этого для каждого активного вокселя создаётся однозначное отображение индекса в координаты ячейки (рис. 2.11).

Стоит заметить, что порядок обхода неважен — главное, чтобы он был одинаков в обоих этапах.

2.4 Рендеринг

2.4.1 Освещение

Освещение в данной работе основано на использовании модели освещения Фонга [9], полагая белый источника света. Метод требует сравнительно мало ресурсов, но большинство оптических явлений игнорируются либо

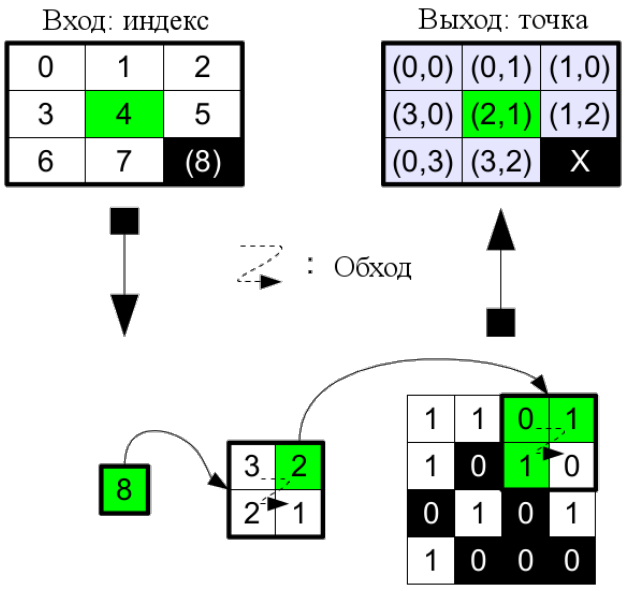

Рисунок 2.11 — Создание списка активных вокселей

рассчитываются с грубым приближением. В освещении по Фонгу интерполируется вектор нормали. Для нахождения вектора нормали в произвольной точке треугольника используют билинейную интерполяцию.

Рассмотрим следующие векторы (рис. 2.12):

— $\hat{\mathbf{N}}$, нормаль к плоскости;
— $\hat{\mathbf{V}}$, направление к наблюдателю;
— $\hat{\mathbf{L}}$, направление к источнику света;
— $\hat{\mathbf{R}}$, отражённый луч источника света;

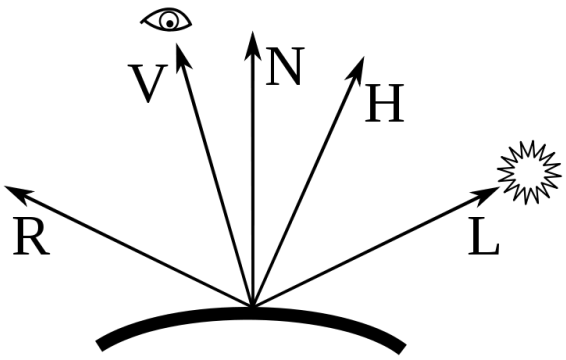

Рисунок 2.12 — Векторы для вычисления освещения

В методе Фонга освещение раскладывается на три составляющие: фоновое, рассеянное и зеркальное освещение.

2.4.1.1 Фоновое освещение

Фоновое освещение — грубое приближение лучей света, рассеянных соседними объектами и затем достигших заданной точки:

$$I_a = k_a c, \tag{2.53}$$

где k_a — константа фонового освещения;
c — цвет поверхности в данной точке.

2.4.1.2 Рассеянное освещение

Рассеянное освещение — основная составляющая освещения:

$$I_d = k_d(\hat{\mathbf{L}} \cdot \hat{\mathbf{N}})c, \tag{2.54}$$

где k_d — константа рассеянного освещения;
c — цвет поверхности в данной точке.

2.4.1.3 Зеркальное освещение

Зеркальное освещение — составляющая освещения, отвечающая за зеркальность и матовость поверхности:

$$I_s = k_s(\hat{\mathbf{R}} \cdot \hat{\mathbf{V}})^\alpha, \tag{2.55}$$

где k_s — константа зеркального освещения;
α — матовость поверхности.

Стоит заметить, что в уравнение не входит цвет поверхности, в отличие от других составляющих. Внешняя поверхность автомобиля является хорошим примером. Рассеянную составляющую обеспечивает краска, в то время как зеркальную — лак. Таким образом, часть лучей отражается, не доходя до краски.

2.4.1.4 Затухание

Затухание — эффект снижения интенсивности источника при его отдалении от поверхности. В реальном мире, интенсивность обратно пропорционально квадрату расстояния:

$$I \propto \frac{1}{d^2},$$

где d — расстояние от источника света до точки поверхности.

Однако для избежания возможного деления на нуль, будем использовать модифицированную версию:

$$a = \frac{1}{1 + d^2},$$

где a коэффициент затухания.

Во-вторых, для контроля над тем, как быстро интенсивность уменьшается с расстоянием, введём дополнительную константу:

$$a = \frac{1}{1 + kd^2}, \tag{2.56}$$

где a — коэффициент затухания;
k — константа затухания.

2.4.1.5 Итоговое уравнение

Таким образом, итоговая формула принимает вид (рис. 2.13):

$$I = k_a c + a\left(k_d(\hat{\mathbf{L}} \cdot \hat{\mathbf{N}})c + k_s(\hat{\mathbf{R}} \cdot \hat{\mathbf{V}})^{\alpha}\right), \tag{2.57}$$

$$a = \frac{1}{1 + kd^2},$$

где I — конечный цвет фрагмента;

c — цвет поверхности в данной точке;

k_a — константа фонового освещения;

k_d — константа рассеянного освещения;

k_s — константа зеркального освещения;

a — коэффициент затухания;

k — константа затухания;

d — расстояние от источника света до точки поверхности;

$\hat{\mathbf{N}}$ — нормаль к плоскости;

$\hat{\mathbf{V}}$ — направление к наблюдателю;

$\hat{\mathbf{L}}$ — направление к источнику света;

$\hat{\mathbf{R}}$ — отражённый луч источника света.

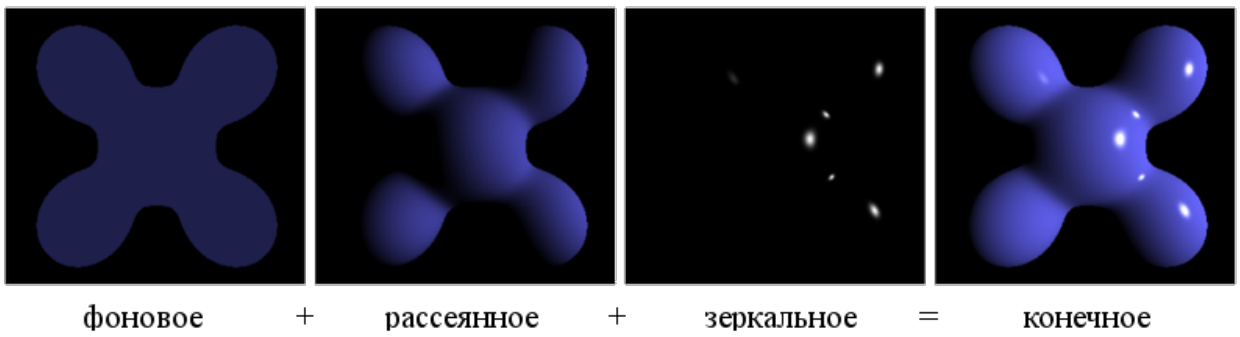

Рисунок 2.13 — Составляющие освещения.

2.4.2 Буфер глубины

Буфер глубины представляет собой двумерный массив, каждый элемент которого соответствует пикселю на экране. Когда GPU рисует пиксель, его удалённость просчитывается и записывается в ячейку буфера глубины. Если пиксели двух рисуемых объектов перекрываются, то их значения глубины сравниваются, и рисуется тот, который ближе, а его значение удалённости сохраняется в буфер. По буферу глубины определяется удалённость от зрителя того или иного объекта трехмерной сцены.

Альтернативные алгоритмы («алгоритм художника» и «двоичное разбиение пространства») используются только при программной отрисовке, так

как аппаратно эффективнее оказывается буфер глубины, поэтому в данной работе не рассматриваются.

2.4.3 Полупрозрачность

Полупрозрачность классически реализуется в два шага: в первую очередь рисуются все непрозрачные объекты с использованием буфера глубины, после чего рисуются полупрозрачные объекты в отсортированном по дальности порядке, причём буфер глубины используется, но не заполняется. При этом, если тест глубины пройден, то цвет получается смешиванием находящегося в буфере и обрабатываемого фрагмента цветов:

$$o = s_a s + (1 - s_a)d \qquad o_a = s_a \tag{2.58}$$

где s — значение цвета обрабатываемого фрагмента;
s_a — alpha-канал обрабатываемого фрагмента;
d — значение находящегося в буфере цвета;
o — значение результирующего цвета;
o_a — alpha-канал результата.

Стоит заметить, что в данной работе полупрозрачностью обладает только поверхность жидкости, а значит порядком отрисовки полупрозрачных объектов можно пренебречь.

3 Технологический раздел

3.1 Упрощение поиска ближайших частиц

Для ускорения поиска соседних частиц активно применяется разбиение пространства [10]. В качестве простого и достаточно эффективного разбиения удобно взять разбиение трёхмерной сеткой.

Тогда индекс ячейки этой сетки для частицы i:

$$\mathbf{c}(\mathbf{r}_i) = \left\lfloor \frac{\mathbf{r}_i - \mathbf{r}_c}{s} \right\rfloor,$$

где $\mathbf{r}_c$ — координаты нулевой ячейки;

$\mathbf{r}_i$ — координаты частицы i;

s — размер ячейки.

Стоит заметить, что в данной работе для симуляции задействована область $0..1$, а значит нулевая ячейка будет располагаться в памяти как первый элемент массива. Поэтому для упрощения расчётов выгодно положить $\mathbf{r}_c = -s$, что позволит избавиться от проверок выхода за пределы сетки.

Таким образом, итоговая формула для вычисления индекса ячейки принимает вид

$$\mathbf{c}(\mathbf{r}_i) = \left\lfloor \frac{\mathbf{r}_i}{s} \right\rfloor + 1, \tag{3.1}$$

Теперь, имея индекс ячейки, необходимо связать ячейку с данной частицей. Однако для экономии вычислительных ресурсов и памяти, было решено упростить физическую модель в пользу упрощения расчетов и хранить средние характеристики частиц для каждой ячейки вместо непосредственно самих частиц. Таким образом, для обхода соседних частиц, достаточно проанализировать «усреднённые частицы» в соседних ячейках.

Размер ячейки для для симуляции может быть вычислен как

$$s(h, n) = \frac{2h}{n} \tag{3.2}$$

где h — радиус сглаживания;

n — кол-во соседей по одной оси $(3, 5, ...)$.

Размер же вокселя для триангуляции задаётся пользователем и напрямую влияет на качество получаемого изображения.

3.2 Сглаживание воксельной сетки

Качество результирующего изображения сильно зависит от размера воксельной сетки. Чем меньше размер вокселя, тем качество выше. Однако при уменьшении размера, количество частиц может не хватать для заполнения отдельных вокселей, что ведёт к образованию дыр и нерегулярности структуры.

Для решения данной проблемы применяется сглаживание сетки: для каждого вокселя применяется сглаживающее трёхмерное ядро, которое чаще всего является некоторой аппроксимацией гауссово ядра сглаживания.

В данной работе при сглаживании для каждого вокселя учитываются шесть прилегающих вокселей:

$$p_i \leftarrow \frac{1}{8}\left(2p_i + \sum_{j=0}^{5} p_j\right), \tag{3.3}$$

где p_i — значение потенциала в текущем вокселе;

p_j — значение потенциала в вокселях, имеющих общую грань с данной.

3.3 Описание алгоритма

Исходя из проведённых в ходе работы исследований и полученных знаний, можно составить полный алгоритм моделирования поведения жидкости и её рендеринга.

3.3.1 Симуляция

Краткая схема работы алгоритма симуляции представлена на рис. 3.1.

Рассмотрим этапы подробнее.

3.3.1.1 Инициализация

а) определить суммарный объём частиц $V = n\frac{m}{\rho}$;

б) создать n частиц и установить их позиции и скорости;

в) создать геометрию сцены (шар и ограничивающий контейнер);

г) инициировать интегратор «чехарды», используя (2.49).

Рисунок 3.1 — Схема алгоритма симуляции.

3.3.1.2 Усреднение позиций и скоростей

Для каждой частицы i:

а) определить занимаемую ячейку, используя (3.1);
б) увеличить сумму позиций в ячейке на позицию частицы $\mathbf{r}_i$;
в) увеличить сумму скоростей в ячейке на скорость частицы $\mathbf{u}_i$;
г) увеличить счетчик частиц в ячейке на 1.

3.3.1.3 Вычисление плотностей

Для каждой частицы i:

а) найти соседние частицы N_i в радиусе h;
б) вычислить плотность ρ_i, используя (2.19).

3.3.1.4 Усреднение плотностей

Для каждой частицы i:

а) определить занимаемую ячейку, используя (3.1);
б) увеличить сумму плотностей в ячейке на плотность частицы ρ_i.

3.3.1.5 Вычисление новых позиций и скоростей

Для каждой частицы i:

а) найти соседние частицы N_i в радиусе h;
б) вычислить давление p_i, используя (2.21);
в) вычислить внутренние силы:

1) вычислить силу разницы давлений $\mathbf{f}_i^p$, используя (2.24);
2) вычислить силу вязкости $\mathbf{f}_i^v$, используя (2.27);
3) $\mathbf{f}_i^{\text{внут}} \leftarrow \mathbf{f}_i^p + \mathbf{f}_i^v$;

г) вычислить внешние силы:
1) вычислить силу гравитации $\mathbf{f}_i^g$, используя (2.29);
2) вычислить нормаль к поверхности $\mathbf{n}_i$, используя (2.31);
3) проверить близость частицы к поверхности, используя (2.33);
4) если частица на поверхности, то
вычислить силу поверхностного натяжения $\mathbf{f}_i^s$, используя (2.32);
5) $\mathbf{f}_i^{\text{внеш}} \leftarrow \mathbf{f}_i^g + \mathbf{f}_i^s$.

д) $\mathbf{F}_i \leftarrow \mathbf{f}_i^{\text{внут}} + \mathbf{f}_i^{\text{внеш}}$;
е) вычислить ускорение $\mathbf{a}_i$, используя (2.17);
ж) вычислить новое значение скорости $\mathbf{u}_i$, используя (2.47);
з) вычислить новое значение позиции $\mathbf{r}_i$, используя (2.48);
и) проверить столкновение с окружением, используя (2.34);
к) если столкновение произошло, то
1) скорректировать позицию $\mathbf{r}_i$, используя (2.45);
2) скорректировать скорость $\mathbf{u}_i$, используя (2.46);
л) аппроксимировать новое значение скорости $\mathbf{u}_i$, используя (2.50).

3.3.2 Триангуляция и рендеринг

Краткая схема работы алгоритмов триангуляции и рендеринга представлена на рис. 3.2.

Рассмотрим этапы подробнее.

3.3.2.1 Нахождение непустых вокселей

Для каждой частицы i:

а) определить занимаемый воксель $\mathbf{c}$, используя (3.1);
б) пометить ячейку $\mathbf{c}$ как непустую.

3.3.2.2 Сглаживание значений

Пусть l — степень сглаживания, заданная пользователем.

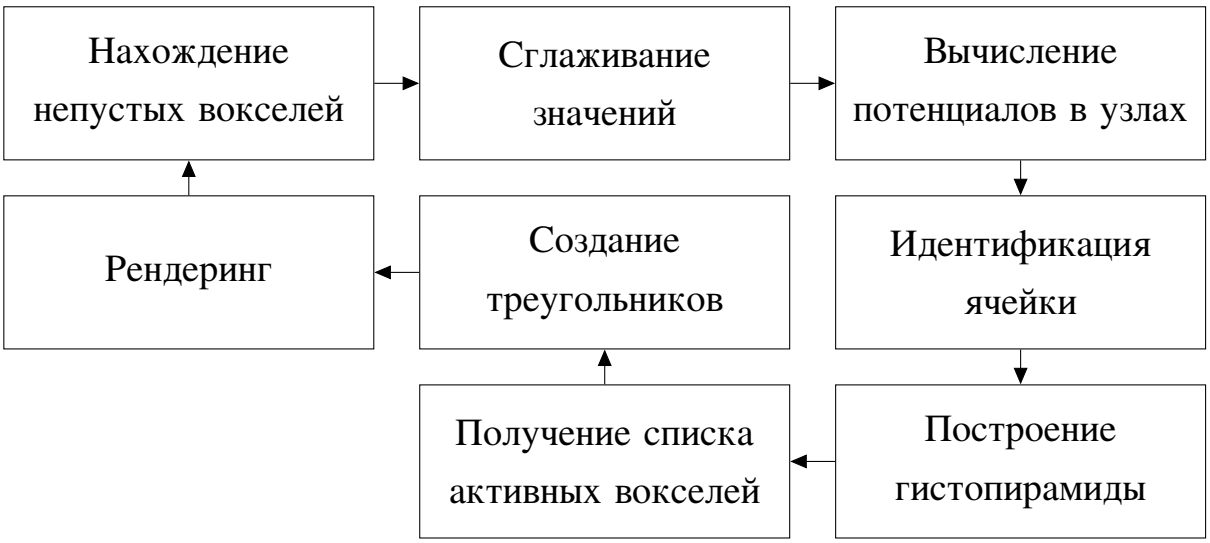

Рисунок 3.2 — Схема триангуляции и рендеринга.

Повторить l раз:

а) Для каждого вокселя $\mathbf{c}$:
   1) вычислить новое значение, используя (3.3).

3.3.2.3 Вычисление потенциалов в узлах

Для каждого узла сетки вокселей $\mathbf{n}$:

а) определить четыре прилежащих вокселя $\mathbf{c}_i$;
б) вычислить потенциал p узла усреднением значений в данных вокселях.

3.3.2.4 Идентификация ячейки

Для каждого вокселя $\mathbf{c}$:

а) определить восемь прилежащих узлов $\mathbf{n}_i$;
б) определить идентификатор комбинации I, используя (2.51).

3.3.2.5 Построение гистопирамиды

Пусть $l = lb(s)$ — количество уровней гистопирамиды, где s — размер сетки вокселей.

Повторить l раз:

а) провести свёртку (рис. 2.10).

3.3.2.6 Получение списка активных вокселей

Пусть n — количество активных вокселей, полученных в результате свёртки сетки вокселей.

Для $i = \overline{0,n-1}$:

а) определить соответствующий воксель (рис. 2.11).

3.3.2.7 Создание треугольников

Имеем:

— I, мн-во одномерных индексов активных вокселей;
— C, мн-во трёхмерных индексов активных вокселей;
— Id, мн-во идентификаторов комбинаций;
— J, мн-во индексов возможных треугольников, $J = \{0,1,2,3\}$;
— K, мн-во индексов вершин, $K = \{0,1,2\}$;
— V, мн-во радиус-векторов вершин;
— A, таблица активных вокселей, $A : I \to C$;
— M, таблица идентификаторов комбинаций, $M : C \to Id$;
— E, таблица рёбер, $E : Id \times J \times K \to V \times V$.

Для каждого активного вокселя:

а) получить воксель $\mathbf{c} \in C$;
б) получить идентификатор комбинации $i \leftarrow M(c)$;
в) для $\forall j \in J, \forall k \in K : (\mathbf{v}_a, \mathbf{v}_b) \leftarrow E(i,j,k)$:
   1) вычислить координаты v вершины, используя (2.52);
   2) прибавить к v координаты ячейки $\mathbf{c}$;
   3) вычислить нормаль n, используя соседние потенциалы p_i.

3.3.2.8 Рендеринг

Для каждого фрагмента каждого треугольника:

а) интерполировать нормаль $\mathbf{n}$;
б) вычислить расстояние до источника света d;
в) вычислить интенсивность фонового освещения I_a, используя (2.53);

г) вычислить интенсивность рассеянного освещения I_d, используя (2.54);

д) вычислить интенсивность зеркального освещенияI_s, используя (2.55);

е) вычислить затухание a, используя (2.56);

ж) инициировать цвет c в зависимости от объекта;

з) вычислить результирующий цвет, используя модель Фонга (2.57).

3.4 Структуры данных

Так как моделирование и рендеринг будет осуществляться на GPU, структуры данных необходимо подбирать соответствующие. Современные графические процессоры на аппаратном уровне оперируют матрицами и векторами чисел с плавающей запятой («float», 32-битные, одинарная точность), а коммуникация с ними производится исключительно с помощью структур, сформированных из этих чисел: векторов, массивов, двумерных таблиц векторов (текстур).

Векторы – одномерные массивы из двух ($vec2$), трех ($vec3$) или четырех ($vec4$) элементов типа $float$. Могут использоваться как для хранения координат, так и для хранения цветов: $vec3$ позволяет хранить один RGB-триплет, где каждая компонента имеет значение в диапазоне $[0, 1]$, $vec4$ – RGBA-значение, где A – непрозрачность в таком же диапазоне.

Матрицы – двумерные массивы из четырех ($mat2$ — 2×2), девяти ($mat3$ – 3×3) или шестнадцати ($mat4$ — 4×4) элементов типа $float$. Расположены в памяти по столбцам, то есть первые 4 элемента — первый столбец, вторые — второй, так далее.

Шар — содержит информацию о положении центра, радиусе, цвете, а также свою полигональную модель, в которую входят нормали и вершины.

Контейнер — содержит вектор положения и размеры по осям.

Камера — содержит вектор позиции, вектор направления и вертикальный вектор.

Информация о **частицах** (скорости, положения и плотности) хранится в параллельных массивах (текстурах), так как так их проще передавать на GPU.

3.5 Используемые технологии

Технология для параллельных вычислений

Так как программы подобного рода отлично поддаются параллелизации, то очевиден выбор GPU и для симуляции и для рендеринга. Для рендеринга выбор невелик: использование OpenGL, DirectX или WebGL (обёртка для работы с OpenGL из браузера, при возможности транслирующая вызовы в DirectX посредством технологии ANGLE). В сфере симуляции выбора больше: CUDA, OpenCL или вычисляющие шейдеры в OpenGL, DirectX, WebGL. CUDA ограничивает использование только на GPU от Nvidia, в то время как драйвера OpenCL недостаточно оптимизированы. В результате была выбрана технология WebGL, поддерживаемая большинством GPU, для работы которой достаточно только браузера.

Язык программирования

Так как WebGL работает из браузера, то в качестве хостового ЯП был выбран EcmaScript 2015 — будущая версия ЯП JavaScript. Однако его поддерживают пока далеко не все браузеры, поэтому исходных код транслируется компилятором babel в Javascript 1.5 — предыдущую версию языка.

В качестве ЯП для GPU выбран GLSL ES — родной для WebGL.

Используемые библиотеки

— dat-gui, библиотека для построения простых пользовательских интерфейсов, пригодных для задач моделирования.

— gl-matrix, высокопроизводительная библиотека с поддержкой SIMD, предоставляющая базовые операции для работы с векторами и матрицами.

Окружение разработчика

В качестве редактора кода был использован Vim на ОС Arch Linux. Сборка проекта осуществляется пакетным менеджером npm — родным инструментом для разработчиков на JS. В качестве системы контроля версий использовался git.

3.6 Пользовательский интерфейс

Приложение выполнено в стиле одностраничного html-документа (рис. 3.3). Пользовательский интерфейс состоит из следующих компонентов:

— Холст, который используется для рендеринга сцены;

— Счётчики (левый верхний угол): счётчик шагов итерации и счётчик кадров рендеринга;

— Кнопки для перезапуска и остановки/запуска симуляции (правый верхний угол);

— Форма для задания параметров моделирования, которые применяются сразу; Для удобства пользователя форма разделена на несколько подразделов, объединяющих связанные параметры.

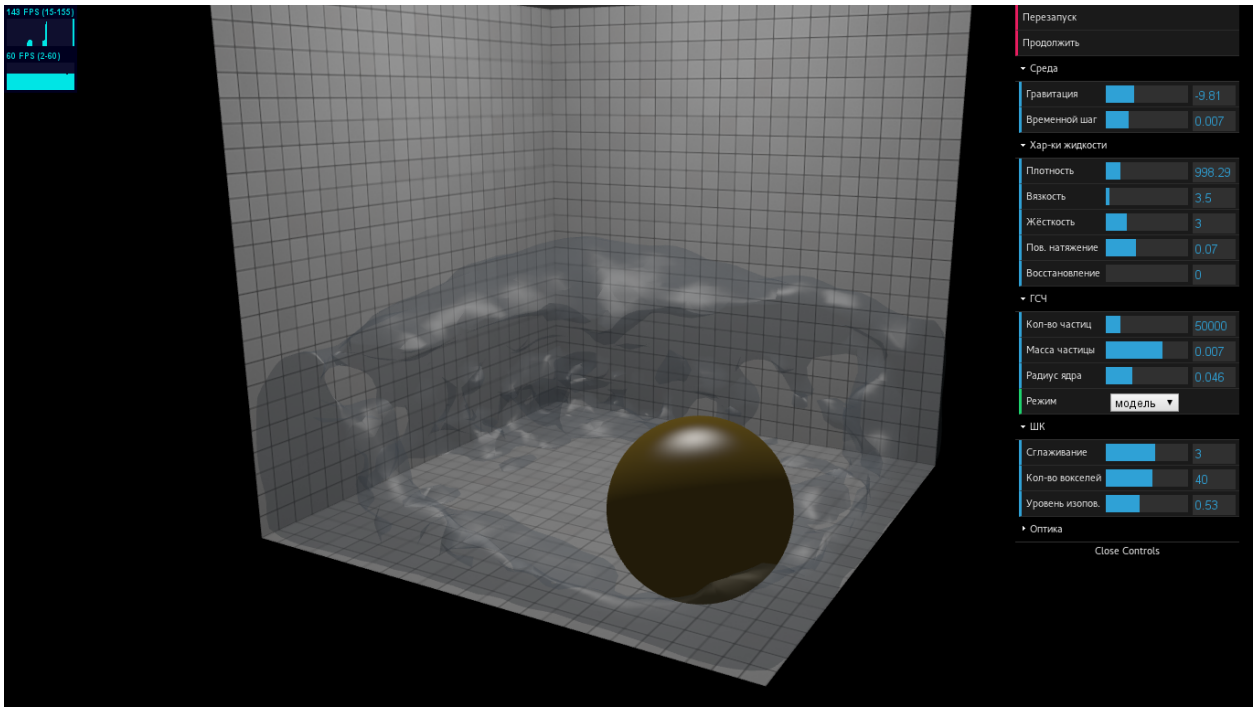

Рисунок 3.3 — Вид приложения.

Для взаимодействия с шаром необходимо нажать на него левой кнопкой мыши и, не отпуская кнопку, передвинуть в нужное место. Для управления камерой необходимо нажать в любое другое место на холсте.

4 Исследовательский раздел

4.1 Производительность

Двумя основными характеристиками, влияющими на производительность, являются время шага симуляции Δt и количество моделируемых частиц N. Замеры производительности проводятся с использованием одной видеокарты[1].

4.1.1 Допустимое время симуляции

Зафиксируем частоту кадров при рендеринге каркаса и исследуем зависимость частоты шагов симуляции от количества частиц (рис. 4.1). Мы можем наблюдать зависимость, близкую к обратной пропорциональности: $\frac{1}{\min \Delta t} \propto \frac{1}{N}$. Отсюда можно сделать вывод, что $\min \Delta t \propto N$.

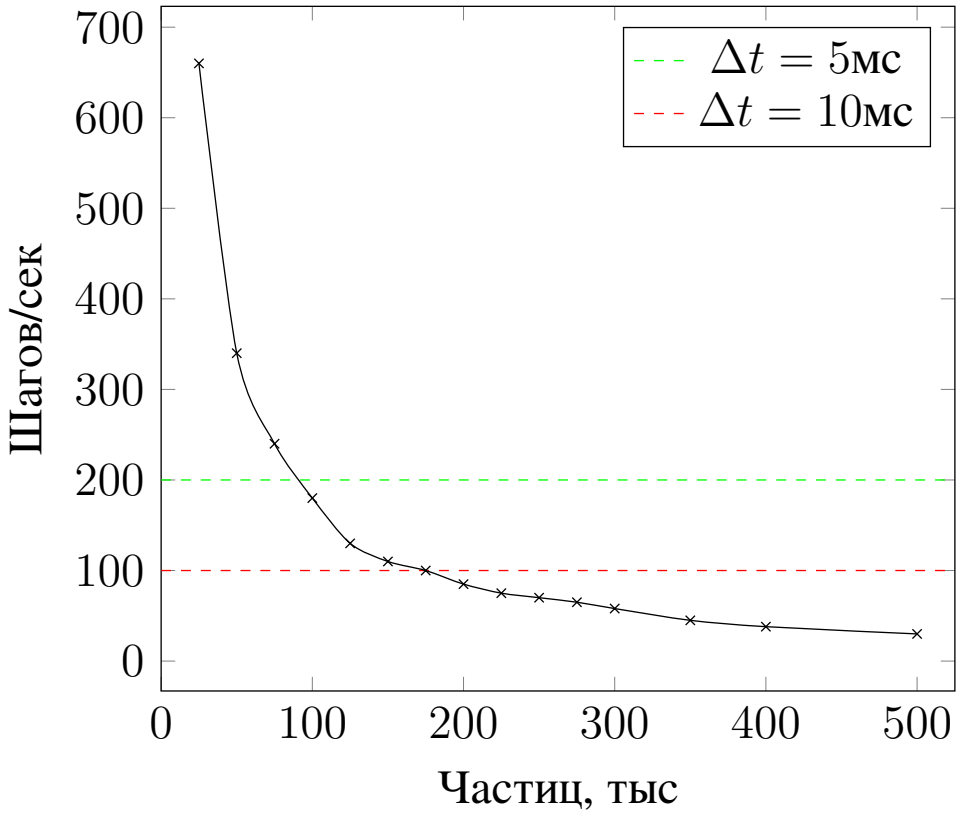

Рисунок 4.1 — Зависимость частоты шагов симуляции от количества частиц при 30к/с рендеринге каркаса.

[1] Nvidia GeForce GTX 750.

4.1.2 Частота кадров симуляции реального времени

Исследуем частоту кадров для симуляции, проводящейся в реальном времени, то есть на один шаг симуляции тратится не более Δt (рис. 4.2). Результаты эксперимента позволяют сделать вывод о пропорциональности количества частиц и частоты кадров: $fps \propto N$.

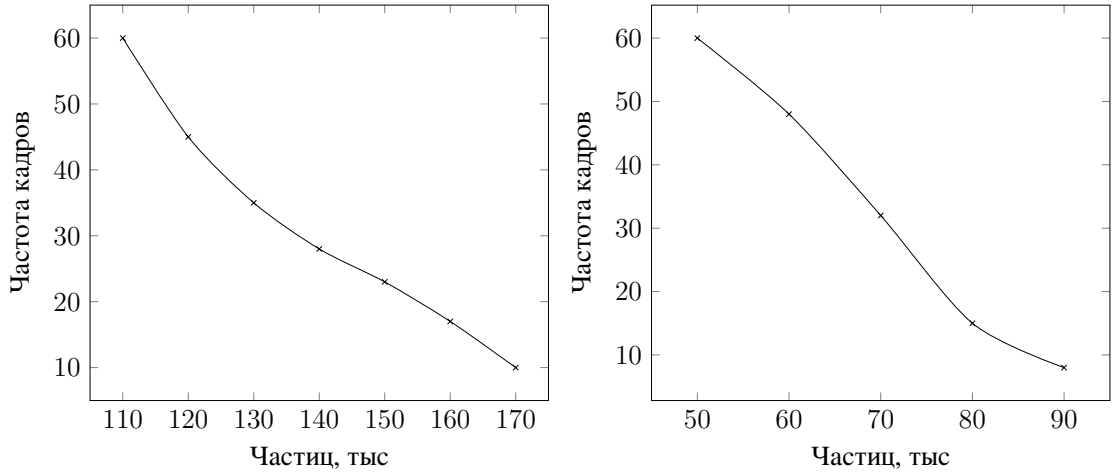

Рисунок 4.2 — Зависимость частоты кадров от количества частиц при $\Delta t = 10$мс и $\Delta t = 5$мс в режиме реального времени.

4.2 Физические явления

4.2.1 Жидкость в условиях нулевой гравитации

В условиях нулевой гравитации ($\mathbf{g} = \mathbf{0}$) жидкость под действием сил поверхностного натяжения стремится принять форму шара (рис. 4.3).

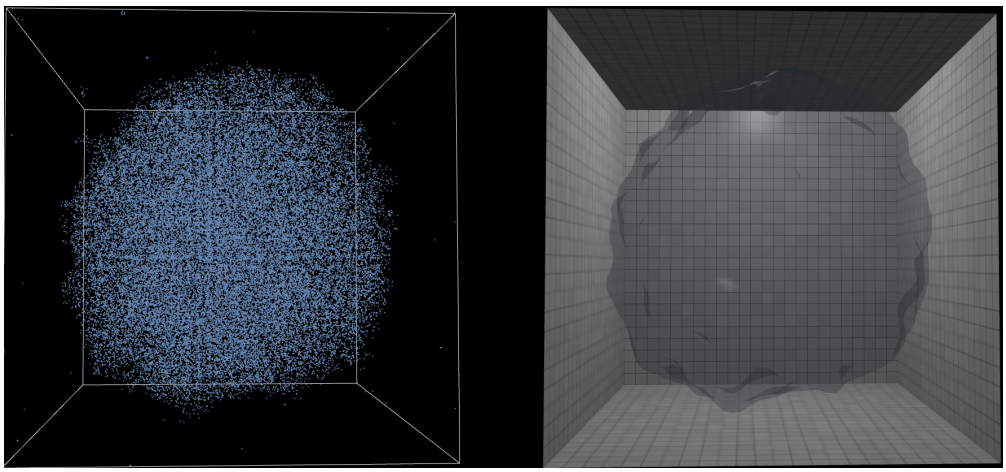

Рисунок 4.3 — Жидкость в условиях нулевой гравитации.

Заключение

В ходе выполнения работы был реализован программный продукт, полностью отвечающий требованиям, изложенным в техническом задании, а именно программа для моделирования поведения жидкости и её взаимодействия с внешним контейнером и шаром, предоставляющая также графический интерфейс и интерактивность для пользователя с целью изменения физических характеристик моделируемой жидкости.

В течение выполнения проекта я провёл анализ существующих методов, позволяющих решить поставленные задачи, на основе проведённого анализа выбрал тот метод, который наиболее подходит для решения поставленных задач, изучил пути оптимизации и ускорения выбранного метода, разработал свой алгоритм решения задачи, построил структуру программы и выделил структуры данных, разработал программу, а также провёл ряд экспериментов и тестов, показавших эффективность разработанного решения.

Список использованных источников

1. *Tao, Terence*. Finite time blowup for an averaged three-dimensional Navier-Stokes equation. — 2014.

2. *Liu, G. R.* Smoothed particle hydrodynamics: a meshfree particle method / G. R. Liu, M. B. Liu. — World Scientific Publishing, 2003.

3. *Monaghan, J.J.* Smoothed Particle Hydrodynamics / J.J. Monaghan // *Ann. Rev. Astron. Astrophys*. — 1992. — Vol. 30. — Pp. 543–74.

4. *Müller, Matthias*. Particle-based Fluid Simulation for Interactive Applications / Matthias Müller, David Charypar, Markus Gross // Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. — SCA '03. — Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2003. — Pp. 154–159.

5. *Desbrun, Mathieu*. Smoothed Particles: A New Paradigm for Animating Highly Deformable Bodies / Mathieu Desbrun, Marie-Paule Gascuel // Proceedings of the Eurographics Workshop on Computer Animation and Simulation '96. — Springer-Verlag New York, Inc., 1996. — Pp. 61–76.

6. *Kelager, Micky*. Lagrangian Fluid Dynamics Using Smoothed Particle Hydrodynamics. — 2006.

7. *Lorensen, William E.* Marching Cubes: A High Resolution 3D Surface Construction Algorithm / William E. Lorensen, Harvey E. Cline // *SIGGRAPH Comput. Graph.* — 1987. — aug. — Vol. 21, no. 4. — Pp. 163–169.

8. GPU point list generation through histogram pyramids: Research Report MPI-I-2006-4-002 / Gernot Ziegler, Art Tevs, Christian Theobalt, Hans-Peter Seidel. — Stuhlsatzenhausweg 85, 66123 Saarbrücken, Germany: Max-Planck-Institut für Informatik, 2006. — June.

9. *Phong, Bui Tuong*. Illumination for Computer Generated Pictures / Bui Tuong Phong // *Commun. ACM*. — 1975. — jun. — Vol. 18, no. 6. — Pp. 311–317.

10. *Harada, Takahiro*. Smoothed Particle Hydrodynamics on GPUs / Takahiro Harada, Seiichi Koshizuka, Yoichiro Kawaguchi // *Proc. of Computer Graphics International*. — 2007. — Pp. 63–70.