Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМ. Н. Э. БАУМАНА (национальный исследовательский университет)

УДК _____
№ госрегистрации _____
Инв. №_____

УТВЕРЖДАЮ

_____
головной исполнитель НИР

_____ _____

«_____» _____ 2019 г.

ОТЧЁТ
О НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ

по теме:

Программа моделирования движения воды с использованием вокселов
(промежуточный)

Руководитель темы _____ А. С. Кострицкий

Москва 2019

РЕФЕРАТ

Отчет содержит 25 стр., 2 рис., 21 источн., 2 прил.

Это пример каркаса расчётно-пояснительной записки, желательный к использованию в РПЗ проекта по курсу РСОИ .

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Визуализация различных явлений становится всё более важной во множестве инженерных областей знаний. Задача объёмного рендеринга имеет большое значение, например, в визуализации данных компьютерной и магнитно-резонансной томографии[11]. Интерактивная 3D симуляция позволяет учёным ясно воспринимать и оценивать результаты собственных исследований.

В настоящее время для этого используются различные вычислительные техники обработки и графического представления экспериментальных данных[6].

В данной работе рассматривается 3D симуляция воды. Симуляция жидкостей, в целом, является примером того, что получаемые о них сведения без соответствующего 3D изображения довольно сложны для человеческого восприятия.

Целью работы является разработка программного продукта для моделирования движения воды с использованием воксельной графики. Для достижения поставленной цели необходимо решить следующие задачи:

— проанализировать существующие методы моделирования движения жидкостей и методы рендеринга с помощью вокселей;

— спроектировать программное обеспечение, симулирующее поведение воды;

— реализовать программу и проверить её работоспособность.

1 Аналитический раздел

В данном разделе производится анализ методов вычислений характеристик жидкостей и их преобразований в графический формат.

Далее рассматриваются идеи применения данных методов к симуляции жидкостей и существующие решения в этой области.

1.1 Физическая модель

Жидкости моделируются как векторное поле скорости жидкости и скалярное поле плотности. Движение задаётся уравнениями Навье-Стокса [1].

Далее рассматривается только движение воды (несжимаемой жидкости) в условиях постоянной температуры.

Тогда уравнения Навье-Стокса в векторной форме принимают следующий вид:

$$\frac{\partial \vec{v}}{\partial t} + \vec{v} \cdot \nabla \vec{v} = \vec{F} - \frac{1}{\rho} \nabla p + \eta \Delta \vec{v}. \tag{1.1}$$

В уравнении 1.1 $\vec{v}$ - скорость частицы воды, t - время, $\vec{F}$ - внешняя удельная сила, p - давление, $\eta = \frac{\mu}{\rho}$ - кинематический коэффициент вязкости, ∇ - оператор Гамильтона, Δ - оператор Лапласа.

Данная физическая модель лежит в основе многих подходов симуляции жидкостей[11]. Их обзор приведён далее.

В статистической физике модель поведения частиц жидкости описывается кинетическим уравнением Больцмана. Данная модель применима для систем, где есть ограничения на малую скорость частиц[12].

1.2 Существующие подходы к симуляции жидкостей

В вычислениях поведения жидкости необходимо представить физическую модель в дискретном виде. Данную проблему решает вычислительная гидродинамика - совокупность теоретических, экспериментальных

и численных методов, предназначенных для моделирования потоковых процессов.

Наиболее распространёнными методами описания характеристик жидкости в вычислительной гидродинамике являются:

— сеточные методы Эйлера;

— метод гидродинамики сглаженных частиц;

— методы, основанные на турбулентности;

— метод решёточных уравнений Больцмана.[11]

Сеточные методы Эйлера являются наиболее простым решением симуляции жидкостей Они заключаются в поиске решения задачи Коши для функций, заданных таблично. Для каждого узла функции уровня жидкости требуется вычисление значений разложений Тейлора в их окрестностям. Решение задачи Коши в данном случае аппроксимирует решение уравнений Навье-Стокса[14].

На основе сеточных методов Эйлера основан способ, который аппроксимирует решение уравнений Навье-Стокса при помощи клеточного автомата[4]. Жидкость представляется в виде трёхмерной сетки. Для каждой "клетки"жидкости в каждом новом поколении вычисляется новое состояние - кинетическая энергия и уровень жидкости - на основе состояний "соседей фон Неймана"этой ячейки жидкости.

Метод гидродинамики сглаженных частиц и методы, основанные на турбулентности, заключаются в выборе размера частицы ("длины сглаживания"), на котором их свойства "сглаживаются"посредством функции ядра или интерполяции, и решения уравнений Навье-Стокса с учётом вязкости и плотности. Это позволяет эффективно моделировать поведение жидкостей, газов и даже использовать в астрофизике[15].

Метод решёточных уравнений Больцмана основан на кинетическом уравнении Больцмана, упомянутом ранее. Этот метод поддерживает многофазные жидкости, наличие теплопроводности и граничные условия на макроскопическом уровне[13].

В данной работе для симуляции воды используется метод клеточного автомата, поскольку является наименее вычислительно сложной аппроксимацией уравнений Навье-Стокса[4].

1.3 Анализ методов визуализации жидкостей

Основными требованиями к методам симуляции жидкостей со стороны компьютерной графики являются визуальная правдоподобность и скорость анимации.

Исходя из этих условий, в компьютерной графике используются специально модифицированные и оптимизированные методы, основанные на указанных ранее. При этом решается задача объёмного рендеринга - на каждый кадр требуется найти проекцию трёхмерного дискретного набора данных о жидкости[16].

В настоящее время существует всего 2 принципиально различных техники объёмного рендеринга: volume ray marching и реконструкция поверхности[11].

1.3.1 Volume ray marching

Данная техника состоит в том, чтобы создать виртуальный экран и далее проследить путь луча до тех пор, пока он проходит сквозь анимируемые объекты. В итоге луч позволяет определить цвет каждого отдельного пикселя. Настоящий алгоритм трассировки лучей имеет значительную вычислительную сложность, поэтому его часто заменяют на правдоподобные аппроксимации[11].

Одним из способов аппроксимации является трассировки лучей путём замены полигонов на вокселы. Это позволяет существенно повышать производительность анимации за счёт возможностей параллелизма и снижения вычислительной сложности за счёт упрощения signed distance function между источником света и гранями воксела[11]. Недостатком данного подхода является то, что для хранения вокселов требуется большее количество памяти, чем для хранения полигонов.

Другим популярным методом является метод срезов объёмных текстур так же, как и воксельный, является математическим упрощением алгоритма трассировки лучей[9]. Данные сцены представляются в виде 3D текстуры. Рендеринг производится с самого дальнего по отношению к наблюдателю слоя текстуры. В итоге видимыми являются ближайшие из частей слоя. В качестве составляющих срезов могут быть использованы как вокселы, так и полигоны. Этот способ оптимизирован аппаратно для некоторых графических процессоров и в этом случае имеет преимущество перед воксельным методом по времени работы.

1.3.2 Реконструкция поверхности

Реконструкцией поверхности, или трёхмерной реконструкцией, называют процесс получения облика реальных объектов.

В качестве входных данных получают множество точек, характеризующее объект. Далее выбирают точки на поверхности объекта так, что получают полигоны, которые имеют необходимую конфигурацию для аппроксимации формы поверхности в её видимой части. Набор результирующих полигонов - реконструированная модель.

Основным алгоритмом, реализующим этот метод, является marching cubes. Он широко используется в компьютерной и магнитно-резонансной томографии[17].

Идея алгоритма состоит в том, что для кубов, находящихся на поверхности объекта, известно, какие точки лежат внутри объекта и какие снаружи. Это позволяет выбрать приближающий полигон так, чтобы его вершины находились на отрезках, соединяющих вершины куба, в примерных точках реального пересечения с объектом. Также можно объединять полигоны на одной поверхности для оптимизации их хранения.

Таким образом, однажды применив алгоритм, можно использовать объект в анимации, если он не изменяет форму поверхности.

Для предметов, теряющих форму, например, жидкостей требуется повторное вычисление реконструированной поверхности. Подобные вычисления имеют значительную сложность, поэтому на практике, как правило, не используются[11]. Как и в случае с ray marching методом, существуют аппаратные ускорители для данного решения[10].

1.3.3 Оптимизации воксельного способа решения задачи объёмного рендеринга

Существуют случаи, в которых система объёмного рендеринга получает на вход трёхмерные данные, в которых есть области, не требующие отрисовки. В подобных ситуациях следует пропускать вычисление результирующего изображения для пустого пространства[2].

Для хранения вокселов можно использовать различные структуры данных. Наиболее простой является трёхмерный массив вокселов. При хранении вокселов в виде массива может требоваться значительное количество памяти, даже если исходная информация достаточно однородная. Поэтому для оптимизации хранения применяют такие структуры данных, как октодерево и binary sparse partitioning дерево[2].

Использование иерахических структур данных имеет следующий недостаток - они статичны, и их использование для моделей, теряющих форму, требует перестроение всего дерева[2].

Поскольку вода теряет свою форму, были предложены оптимизации не для хранения вокселов, а для обработки лучей и самих данных симуляции. Данные оптимизации основаны на принципах обработки чисел с плавающей запятой на аппаратном обеспечении, а также на возможности ускорения вычислений за счёт параллелизма[11].

1.3.4 Вывод

В данной работе используется метод реконструкции поверхности, который используется совместно с ray marching подходом. В качестве

способа реконструкции используется оптимизация marching cubes для поиска больших прямоугольников на воксельных поверхностях.

В качестве структуры данных для хранения вокселов предлагается использовать хеш-таблицу, в которой хешами являются координаты воксела. Если воксел присутствует в сцене, то для него найдётся соответствующее логическое значение из хеш-таблицы. Данный способ позволяет избежать хранения в памяти пустого пространства сцены.

1.4 Анализ алгоритмов визуализации вокселов

Ранее рассматриваются методы для описания сцены из вокселов, которая представляет собой результат симуляции воды. Для их визуализации выбран метод реконструкции поверхности совместно с ray marching подходом. В данном методе предусмотрена возможность использования множества стандартных алгоритмов компьютерной графики для удаления невидимых линий и поверхностей, а также закрашивания [11]. Далее производится анализ этих алгоритмов визуализации.

1.4.1 Анализ алгоритмов удаления невидимых линий и поверхностей

В данном подразделе рассматриваются алгоритмы для удаления невидимых линий и поверхностей, пригодные для использования в ранее указанном методе:

— алгоритм с использованием Z-буферизации;

— алгоритм с использованием ray marching и signed distance function.

1.4.1.1 Алгоритм с использованием Z-буферизации

Далее описывается суть алгоритма с использованием Z-буферизации.

а) Создаётся двумерный массив - Z-буфер, каждый элемент которого соответствует пикселю изображения.

б) Полученную с помощью реконструкции поверхности триангулированную сетку разбивают на треугольники и находят их проекции на картинную плоскость.

в) Для каждого спроецированного треугольника вычисляется объемлющий прямоугольник.

г) Для каждого пикселя внутри объемлющего прямоугольника вычисляется значение глубины на основе координат вершин спроецированного треугольника.

д) Если значение Z-буфера для данного пикселя меньше, чем значение глубины в нём, то в Z-буфер записывается данное значение глубины. В буфер изображения записывается данный пиксель.

На выходе алгоритма - буфер изображения, каждый пиксель которого окрашен в цвет ближайшего треугольника.

Достоинства алгоритма:

— возможна параллельная обработка треугольников;

— не требуется сортировка вокселов;

— возможно аппаратное ускорение для чтения и записи из Z-буфера.

Недостатком является значительное использование памяти, но для современных вычислительных систем это часто оказывается приемлемым[18].

1.4.1.2 Алгоритм с использованием ray marching и signed distance function

Для данного алгоритма требуется задание функции для каждого объекта сцены, значение которой положительно, если выбранная точка находится снаружи объекта сцены, и отрицательно, если выбранная точка находится внутри объекта сцены.

Далее для каждого луча последовательно выбирают точки до тех пор, пока значение signed distance function положительно. Если её значение стало отрицательным, то найдено пересечение, следовательно, данную точку можно записывать в результат.

Достоинства алгоритма:

— возможно определение реалистичного цвета поверхности на основе пройденного расстояния луча;

— возможно использование многих источников света;

— возможен обратный raymarching (от объектов к картинной плоскости), в котором допустима параллельная обработка лучей.

Недостатки алгоритма:

— зависит от выбора signed distance function;

— вычислительно сложнее, чем алгоритм с использованием Z-буферизации[9].

1.4.1.3 Вывод

В результате выбран алгоритм с использованием Z-буферизации, поскольку временные затраты на вычисление расстояния в ray marching подходе больше.

1.4.2 Анализ алгоритмов закрашивания

1.4.2.1 Закраска по Ламберту

1.4.2.2 Закраска по Гуро

1.4.2.3 Закраска по Фонгу

1.4.2.4 Вывод

1.5 Вывод

Исходя из полученных сведений о достоинствах и недостатках каждого из методов объёмного рендеринга в дальнейшем рассматривается реализация с помощью вокселей. Она обеспечивает возможность изменения формы жидкости и наиболее быструю из предложенных техник трассировки лучей.

2 Конструкторский раздел

В данном разделе описано проектирование метода рендеринга воды с помощью вокселов с указанием соответствующих схем алгоритмов.

2.1 Архитектура приложения

Исходя из поставленной цели, разработанное приложение должно решать задачу синтеза сложного динамического изображения.

Эту задачу принято разделять на следующие этапы:

— разработка трёхмерной математической модели синтезируемой визуальной обстановки;

— задание положения наблюдателя, картинной плоскости, размеров окна вывода, значений управляющих сигналов;

— определение операторов, осуществляющих пространственное перемещение объектов визуализации;

— преобразования координат объектов в координаты наблюдателя;

— отсечение объектов сцены по границам пирамиды видимости;

— вычисление двумерных перспективных проекций объектов на картинную плоскость;

— удаление невидимых линий и поверхностей при заданном положении наблюдателя;

— закрашивание и затенение видимых объектов сцены;

— вывод полученного полутонового изображения на экран растрового дисплея.

Далее приводятся решения для каждого этапа синтеза изображения.

2.1.1 Модуль

2.2 Вывод

3 Технологический раздел

В данном разделе описаны требуемые средства и подходы к реализации ПО по ранее указанным методам.

3.1 Требования к программному обеспечению

Разработанное ПО должно моделировать движение воды с использованием вокселов.

Моделирование движения должно осуществляться с использованием операций переноса, масштабирования и поворота.

3.2 Используемые технологии

Для реализации ПО выбран язык Clojure. Данный язык программирования является компилируемым и, одновременно с этим, динамическим. Clojure - преимущественно язык функционального программирования, который поддерживает нативный доступ к Java фреймворкам [19].

Данные свойства языка позволяют вести разработку приложений с помощью REPL (Read-Eval-Print Loop), модифицируя их во время выполнения. Это может быть полезно для добавления новых функциональностей в программу без её повторной сборки [20].

Для создания пользовательского интерфейса используется Java библиотека Swing. Swing предоставляет широкий набор компонентов для создания графического пользовательского интерфейса. Данные компоненты полностью реализованы на Java, поэтому их внешний вид не зависит от платформы.

Для организации проекта на Clojure и для автоматизации сборки используется Leiningen. Все конфигурации для сборки приложения, запуска REPL и описания зависимостей управляются с помощью этого модуля[21].

3.3 Листинги кода

В листинге 3.3 указана конфигурация сборки проекта на языке Clojure.

```
(defproject computer-graphics-coursework-backend "0.1.0-SNAPSHOT"
  :description "Pudov's computer graphics coursework"
  :url "https://github.com/DPudov"
  :license {:name "Eclipse Public License"
            :url  "http://www.eclipse.org/legal/epl-v10.html"}
  :dependencies [[org.clojure/clojure "1.10.0"]
                 [seesaw "1.5.0"]]
  :repl-options {:init-ns computer_graphics_coursework_backend.core}
  :main computer_graphics_coursework_backend.core
  :profiles {:uberjar {:aot :all}}
  :java-cmd "/usr/lib/jvm/java-1.11.0-openjdk-amd64/bin/java")
```

3.4 Вывод

4 Исследовательский раздел

В данном разделе проводится апробация разработанной программы.

4.1 Примеры использования

4.2 Выводы

ЗАКЛЮЧЕНИЕ

В результате проделанной работы стало ясно, что ничего не ясно...

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Тирский Г. А. Большая российская энциклопедия. Том 21. — Большая российская энциклопедия, 2012.

2. Bautembach Dennis. Animated Sparse Voxel Octrees // UNIVERSITY OF HAMBURG Department of Informatics. — 2011.

3. Zadick Johanne, Kenwright Benjamin, Mitchell Kenny. Integrating Real-Time Fluid Simulation with a Voxel Engine // The Computer Games Journal. — 2016.

4. The game of flow - cellular automaton-based fluid simulation for realtime interaction / Christian Heintz, Moritz Grunwald, Sarah Edenhofer et al. — 2017. — 11. — P. 1–2.

5. Premoze S., Ashikhmin M. Rendering Natural Waters. — Hong Kong, 2000. — P. 22–30.

6. Magnetic Resonance Imaging: Physical Principles and Sequence Design / Robert W. Brown, Y.-C. Norman Cheng, E. Mark Haacke et al. — John Wiley Sons, 2014. — P. 976.

7. VanderHart Luke, Neufeld Ryan. Clojure Cookbook: Recipes for Functional Programming. — O'Reilly Media, Inc., 2014.

8. Lombard Yann. Realistic Natural Effect Rendering: Water I. — 2004. — Access mode: `https://www.gamedev.net/articles/programming/graphics/realistic-natural-effect-rendering-water-i-r2138/`.

9. Wong Jamie. Ray marching and signed distance functions. — 2016. — 7. — Access mode: `http://jamie-wong.com/2016/07/15/ray-marching-signed-distance-functions/`.

10. NVIDIA. NVIDIA® GVDB Voxels. — 2018. — Access mode: `https://developer.nvidia.com/gvdb`.

11. Ash Michael. Simulation and Visualization of a 3D Fluid : Master's thesis / Michael Ash ; Université d'Orléans. — 2005.

12. Мякишев Г. Я. Кинетическое уравнение Больцмана. — Режим доступа: https://www.booksite.ru/fulltext/1/001/008/061/212.htm.

13. Баранов Василий. Моделирование гидродинамики: Lattice Boltzmann Method. — 2013. — Режим доступа: https://habr.com/ru/post/190552/.

14. Асламова В. С., Колмогоров А. Г., Сумарокова Н. Н. Вычислительная математика. Часть вторая: Учебное пособие для студентов дневного и заочного обучения технических и химико-технологических специальностей. — Ангарская государственная техническая академия, 2005. — С. 94.

15. Bate Matthew R., Bonnell Ian A., Bromm Volker. The Formation of Stars and Brown Dwarfs and the Truncation of Protoplanetary Discs in a Star Cluster. — 2002. — Access mode: https://www.ukaff.ac.uk/starcluster/.

16. Lacroute Philippe, Levoy Marc. Fast Volume Rendering Using a Shear-Warp Factorization of Viewing Transformation // Computer Graphics Proceedings, Annual Conference Series. — 1994.

17. Anderson Ben. An Implementation of the Marching Cubes Algorithm. — 2005. — Access mode: http://www.cs.carleton.edu/cs_comps/0405/shape/marching_cubes.html.

18. Bansall Sahil. Z-Buffer or Depth-Buffer method. — 2017. — Access mode: https://www.geeksforgeeks.org/z-buffer-depth-buffer-method/.

19. Hickey Rich. The Clojure Programming Language. — Access mode: https://clojure.org/.

20. Hickey Rich. Programming at the REPL: Introduction. — Access mode: https://clojure.org/guides/repl/introduction.

21. Hagelberg Phil. Leiningen. — Access mode: https://leiningen.org/.

ПРИЛОЖЕНИЕ А

КАРТИНКИ

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Рисунок А.1 — Картинка в приложении. Страшная и ужасная.

ПРИЛОЖЕНИЕ Б

ЕЩЕ КАРТИНКИ

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Рисунок Б.1 — Еще одна картинка, ничем не лучше предыдущей. Но надо же как-то заполнить место.