Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика, искусственный интеллект и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОЙ РАБОТЕ

НА ТЕМУ:

«Визуализация воды с использованием вокселей»

2023 г.

СОДЕРЖАНИЕ

1 Введение

Термин "Компьютерная графика"обозначает любое использование компьютера для создания и манипуляции изображениями. Задачи и области применения компьютерной графики сложно охарактеризовать в силу их большого колчичества, однако в общем случае можно выделить следующие направления:

- **Моделирование** – занимается математическим описанием формы и внешнего вида в формате, пригодным для использования в компьютере;
- **Рендеринг** или **Отрисовка** – термин, заимствонный из изобразительного искусства, обозначающий создание изображений из компьютерных моделей;
- **Анимация** – это техника создания иллюзии движения используя последовательность изображений [1].

Рендеринг – это основная составляющая компьютерной графики. На самом высоком уровне абстракции рендеринг является процессом преобразования описания трехмерной сцены в изображение. Алгоритмы для создания анимации, геометрического моделирования, нанесения текстур и других областей компьютерной графики должны проходить через некий процесс рендеринга, чтобы их результаты могли быть видны на изображении. Рендеринг стал повсеместным: от кино до игр и далее, он открыл новые горизонты для творческого выражения, развлечения и визуализации.

В первые годы развития этой области исследования в графическом рендеринге сосредоточивались на решении фундаментальных проблем, таких как определение, какие объекты видны из определенной точки обзора. По мере нахождения эффективных решений для этих проблем и доступности более богатых и реалистичных описаний сцен благодаря продолжающемуся прогрессу в других областях графики, современный рендеринг начал включать идеи из широкого спектра дисциплин, включая физику и астрофизику, астрономию, биологию, психологию и изучение восприятия, а также чистую и прикладную математику. Междисциплинарный характер рендеринга является одной из причин, почему это такая увлекательная область исследований.

В последние годы активно стала развиваться фотореалистичная компьютерная графика. Это дисциплина находится на стыке физики и классической

компьютерной графики. Высокий реализим изображений требует больших вычислительных мощностей для их получения, что заставляет разработчиков постоянно искать новые, более эффективне способы рендеринга.

Фотореалистичная компьютерная графика теперь повсеместно используется, включая такие области как развлечения, в частности, кино и видеоигры, дизайн продуктов и архитектура. За последнее десятилетие широкое распространение получили физически ориентированные методы визуализации, где точное моделирование физики рассеяния света является основой синтеза изображений. Эти подходы обеспечивают как визуальный реализм, так и предсказуемость [2].

Цель данной работы – реализовать программу для построения реалистичных изображений трехмерных воксельных сцен в реальном времени, с возможностями физически-корректной визуализации воды.

Чтобы достигнуть поставленой цели, требуется решить следующие задачи.

1. Описать структуру трехмерной сцены, включая объекты, из которых состоит сцена и их материалы, и определить способ задания исходных данных;

2. Выбрать или разработать алгоритмы компьютерной графики, позволяющие визуализировать трехмерную воксельную сцену в реальном времени;

3. Реализовать выбранные и адаптированные алгоритмы построения трехмерной сцены;

4. Исследовать возможности улучшения производительности программы и повышения сложности задаваемых сцен.

2 Аналитический раздел

В данном разделе будут рассмотрены возможножсти отрисовки трехмерных изображений в реальном времени, проанализированы особенности трехмерной воксельной графики; будет выбран метод решения поставленной задачи.

2.1 Выбор оборудования

Построение трехмерных изображений – вычислительно сложная, и в то же время, чрезвычайно параллельная задача [3]. Такие особенности стали причиной появляния специального оборудования для выполнения популярных задач компьютерной графики. Такие вычислительные устройства называются Графическими процессорами (GPU), хотя их применение не ограничивается вычислениями графических данных. Появление таких устройств вызвано ограничениями в дизайне центральных процессоров. Они используются для последовательного выполнения кода с большим числом условных переходов, на небольшом числе ядер при симметричной многопроцессорности (SMP). И несмотря на развитие SIMD парадигмы, позволяющей обрабатывать большее число данных, чем позволяет последовательное исполнения, ЦПУ не смогли стать универсальными устройствами для рендеринга. Особенностью графических сопроцессоров ялвяется большое число ядер, способных эффективно выполнять тысячи вычислений параллельно [4].

Несмотря на большие возможности параллельного выполнения вычислительных задач, графические сопроцессоры имеют органиченные возможности условного выполнения программ, в частности, рекурсивных, малую скорость памяти, и возможности взаимодействия с ней.

Такие особенности породили целый ряд алгоритмов, оптимизированных для выполняения на графических сопроцессорах, которые отличаются от аналогичных для процессоров общего назначения. В частности, графические сопроцессоры могут иметь специальные аппаратные блоки для ускоренного выполнения алгоритмов трассировки лучей, и все из них имеют аппартное ускорение алгоритма Z-буффера. [5]

Из-за развития графических сопроцессоров, использование ЦПУ для реализации графических вычислений устарело. Графические сопроцессоры предоставляют более широкие возможности для проведения операций, типич-

ных для компьютерной графики.

2.2 Выбор алгоритма отрисовки

123

2.2.1 Алгоритм, использующий Z-буффер

123 [6]

2.2.2 Трассировка лучей

Почти все системы фотореалистичной рендеринга основаны на алгоритме трассировки лучей. Трассировка лучей на деле очень простой алгоритм; он основан на отслеживании пути луча света через сцену, по мере его взаимодействия и отражения от объектов в окружающей среде. Несмотря на то, что существует множество способов реализовать алгоритм трассировки лучей, все такие системы должны включать в себя и симулировать следующие объекты и феномены:

- Камеры: Модель камеры определяет, как и откуда просматривается сцена, включая то, как изображение сцены записывается на сенсоре. Многие системы рендеринга генерируют оптические лучи, начинающиеся в камере, которые затем прослеживаются в сцене.

- Пересечение лучей и объектов: Нам необходимо точно определить, где заданный луч пересекает заданный геометрический объект. Кроме того, нам нужно определить некоторые свойства объекта в точке пересечения, такие как нормаль поверхности или его материал. Большинство трассировщиков лучей также имеют некоторую возможность проверки пересечения луча с несколькими объектами, обычно возвращают ближайшее пересечение по лучу.

- Источники света: Без освещения бессмысленно визуализировать сцену. Трассировщик лучей должен моделировать распределение света по всей

сцене, включая не только местоположение самих источников света, но и способ, которым они распространяют свою энергию в пространстве.

- Видимость: Чтобы знать, может ли заданный источник света передавать энергию в точку поверхности, нам нужно знать, есть ли непрерывный путь от точки до источника света. К счастью, на этот вопрос легко ответить в трассировщике лучей, так как мы просто можем построить луч от поверхности до источника света, найти ближайшее пересечение луча с объектом и сравнить расстояние пересечения с расстоянием до источника света.

- Рассеяние на поверхности: Каждый объект должен предоставить описание своего вида, включая информацию о том, как свет взаимодействует с поверхностью объекта, а также о характере перераспределенного (или рассеянного) света. Модели рассеивания на поверхности обычно параметризуются таким образом, чтобы можно было смоделировать разнообразные внешние виды.

- Непрямое распространение света: Поскольку свет может достигать поверхности после отражения от других поверхностей или прохождения через них, обычно необходимо проследить дополнительные лучи, исходящие из поверхности, чтобы полностью учесть этот эффект.

- Распространение лучей: Нам нужно знать, что происходит с светом, распространяющимся вдоль луча по мере его прохождения через пространство. Если мы отображаем сцену в вакууме, энергия света остается постоянной вдоль луча. Хотя истинные вакуумы необычны на Земле, для многих сред они являются разумным приближением. Для отслеживания лучей через туман, дым, атмосферу Земли и т. д. доступны более сложные модели.

[2]

Трассировка лучей, являясь крайне простым алгоритмом, предоставляет несчислимые возможности для расширения. Используя трассировку лучей можно добавлять визуализацию физических явлений, вроде отражений, в сцены, отрисованные при помощи Z-буффера. Такой подход применяется в

высокопроизводительных графических приложениях, работающих в реальном времени – при реализации видеоигр.

С другой стороны, алгоритм трассировки лучей может использоваться для создания фотореалистичных изображений [2]. Такие трассировщики используются при создании спецэффектов к фильмам, анимационных картин и др.

Алгоритм кеширования обратной репроекции

Выполнение пиксельных шейдеров потребляет все большую часть вычислительного бюджета для приложений в реальном времени. Однако, значительная временная согласованность в видимых поверхностных областях, условиях освещения и расположении камеры позволяет повторно использовать вычислительно интенсивные расчеты освещения между кадрами, что позволяет достичь значительного повышения производительности при небольшом снижении визуального качества. Кэширование на основе обратной репроекции позволяет пиксельным шейдерам сохранять и повторно использовать расчеты, выполненные в видимых точках поверхности. Такой подход обеспечивает значительное повышение производительности для многих распространенных эффектов в реальном времени, включая предварительно вычисленные глобальные эффекты освещения, стереоскопическую отрисовку, движущийся размытый фон, глубину резкости и теневую картографию. [7]

2.3 Выбор алгоритма обхода воксельной сцены

Воксель (аналогично Пиксель) – способ представления геомтерии, альтернативный типичному полигональному. Воксель представляет собой некоторое значение на регулярной решетке в трехмерном пространстве. Воксели часто испольуются при анализе медицинских и научных данных.

В интерактиных графических приложениях, выполняющих отрисовку в реальном времени, воксели редко применялись в качестве основного геометрического примитива. Это связано с тем, что графические сопроцессоры были специально оптимизированы для работы с полигонылными данными, и они были более простым способом достичь требуемого качества изображения.

В последние годы замечается тенденция повышения популярности вок-

селей в интерактивных приложениях. Это связано с повышением мощности вычислительной техники, позволяющей выполнять ранее невозможные вычисления на пользовательских компьютерах. Воксельная графика позволяет достигать большей детализации, чем возможно при использовании полигонов.

Главная проблема воксельной графики – большое число затрачиваемой памяти и медленный доступ к ней. Поэтому все воксельны алгоритмы фокусируются на ускорении доступа к индивидуальным вокселям, для использования в алгоритмах отрисовки. Рассмотрим и выберем алгоритм обхода воксельной сцены и структуру данных хранения вокселей.

2.3.1 Алгоритм быстрого обхода вокселей для трассировки лучей

123 [8]

2.3.2 KD-дерево

123 [9]

2.3.3 Разреженное воксельное октодерево

123 [10]

2.4 Выбор модели освещения

123

2.4.1 Простая закраска

123 [6]

2.4.2 Модель освещения Фонга

123 [11]

2.4.3 Физически-корректная модель

123 [2]

Микрогранные модели
123 [2] [**MMfRrRS**]

2.5 Вывод

Было принято решение выполнять разработку для выполнения на графическом сопроцессоре, поскольку это позволяет выполнять отрисовку значительно более эффективно, чем на ЦПУ.

В качестве алгоритма трассировки была выбрана обратная трассировка лучей с помощью метода Монте-Карло, поскольку он имеет лучшие возможности для получния физически-корректных изображений, чем алгоритм, использующий Z-буффер. Для оптимизации отрисовки был выбран способ кеширования обратной репроекции.

В качестве модели освещения была выбрана физически-корректная модель с использованием микрограней с моделью GGX, поскольку такой вариант предоставляет лучший вариант получения реалистичных изображений, чем другие модели освещения и модель микрограней Бекерманна [12].

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. *Shirley P.*, *Marschner S.* Fundamentals of Computer Graphics. — 3rd. — USA : A. K. Peters, Ltd., 2009. — ISBN 1568814690.
2. *Pharr M.*, *Jakob W.*, *Humphreys G.* Physically Based Rendering: From Theory to Implementation (3rd ed.) — 3rd. — San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 10.2016. — С. 1266. — ISBN 9780128006450.
3. *Foster I.* Designing and Building Parallel Programs: Concepts and Tools for Parallel Software Engineering. — 75 Arlington Street, Suite 300 Boston, MAUnited States : Addison-Wesley Longman Publishing Co., 01.1995. — ISBN 9780201575941.
4. *Fatahalian K.*, *Houston M.* A Closer Look at GPUs. — 2008.
5. Ray Tracing In Vulkan / D. Koch [и др.]. — 2020. — Дек.
6. *Роджерс Д.* Алгоритмические основы машинной графики. — 1989.
7. Accelerating Real-Time Shading with Reverse Reprojection Caching / D. Nehab [и др.]. — 2007. — Авг.
8. *Amanatides J.*, *Woo A.* A Fast Voxel Traversal Algorithm for Ray Tracing // Proceedings of EuroGraphics. — 1987. — Авг. — Т. 87.
9. *Foley T.*, *Sugerman J.* KD-Tree Acceleration Structures for a GPU Raytracer. — 2005.
10. *Laine S.*, *Karras T.* Efficient Sparse Voxel Octrees – Analysis, Extensions, and Implementation. — 2010.
11. *Phong B. T.* Illumination for Computer Generated Pictures. — 1975.
12. Microfacet Models for Refraction through Rough Surfaces / B. Walter [и др.]. — 2007.